SEQUENT

DYNIX/ptx Operations Guide: sysadm

1003-48607-00

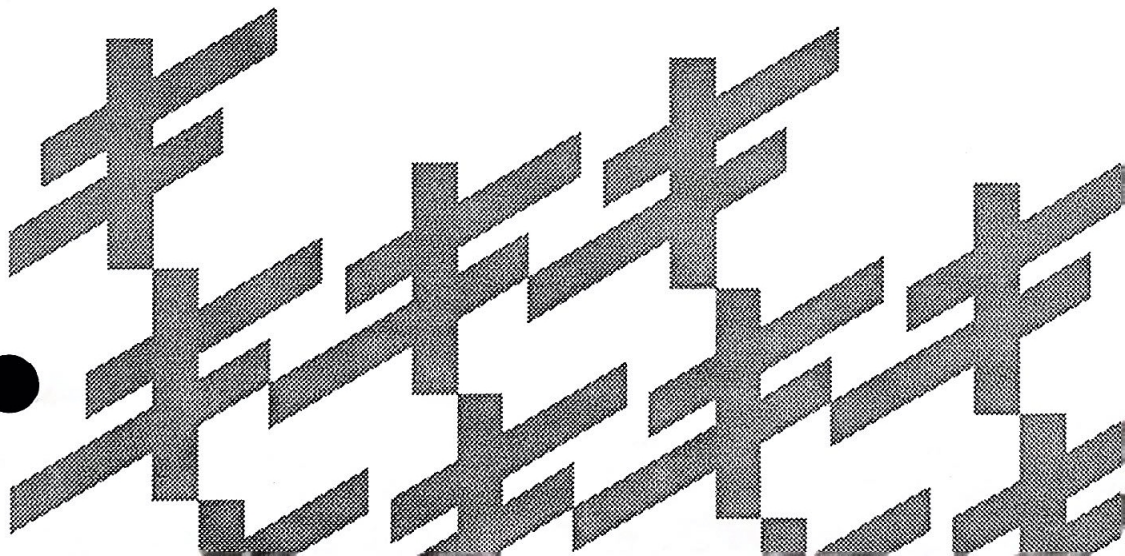
SEQUENT

THIS SHEET LIFTER
MUST BE PLACED ON
TOP OF CONTENTS

GRAPHIC PRODUCTS
Chicago, Illinois 60609

DYNIX/ptx®
Operations Guide: sysadm

1003-48607-00
Sequent Computer Systems, Inc.



DYNIX and DYNIX/ptx are registered trademarks of Sequent Computer Systems, Inc.

UNIX is a registered trademark of AT&T.

Section 4.5.3 of this manual is taken from the article "Fsck - the UNIX File System Check Program," by Marshall Kirk McKusick and T.J. Kowalski. Revised July 16, 1985. Copyright © 1986, Regents of the University of California.

Copyright © 1989 by Sequent Computer Systems, Inc. All rights reserved. This document may not be copied or reproduced in any form without permission from Sequent Computer Systems, Inc. Information in this document is subject to change without notice.

Printed in the United States of America.

Contents

About This Guide

Using sysadm

Chapter P1. Identification and Security Procedures

P1.1 Set Time and Date P1-1
P1.2 Establish or Change System and Node Names P1-3
P1.3 Assign Passwords to Administrative Logins P1-5

Chapter P2. User Services Procedures

P2.1 Add Users or Groups P2-1
P2.2 Modify User Information P2-4
P2.3 Delete Users or Groups P2-7
P2.4 List Users or Groups P2-9
P2.5 Broadcast Messages to All Users P2-11

Chapter P3. Run Level Procedures

P3.1 Powerdown P3-1
P3.2 Shutdown to Single-User P3-3
P3.3 Return to Multiuser P3-4

Chapter P4. Filesystem Administration Procedures

P4.1 Mount or Unmount Filesystems P4-2
P4.2 Maintain Filesystems P4-3
P4.3 Filesystem Backup and Restore P4-6

Chapter P5. LP Print Service Procedures

P5.1 Stop the Print Service P5-1
P5.2 Restart the Print Service P5-2
P5.3 Set Up the Print Service P5-3
P5.4 Set Up Forms P5-11
P5.5 Set Up Filters P5-18

Chapter P6. TTY Management Procedures

P6.1 Check TTY Line Settings P6-1
P6.2 Create TTY Line Settings P6-4
P6.3 Modify TTY Line Characteristics P6-6

Chapter P7. Basic Network Procedures

P7.1 Set Up Basic Networking Files P7-1
P7.2 Basic Networking Maintenance P7-14

P7.3 Basic Networking Debugging P7-18

Chapter 1. System Security
1.1 Introduction 1-1
1.2 Security Guidelines 1-1
1.3 Logins and Passwords 1-2

Chapter 2. User Services
2.1 Introduction 2-1
2.2 Login Administration 2-1
2.3 The User Environment 2-6
2.4 User Communications Services 2-10
2.5 Anticipating User Requests 2-12

Chapter 3. System Operations and Run Levels
3.1 Introduction 3-1
3.2 Operating Levels 3-6

Chapter 4. Filesystem Administration
4.1 Introduction 4-1
4.2 Filesystems 4-1
4.3 Maintaining Filesystems 4-11
4.4 Filesystem Backup and Restore 4-15
4.5 Checking Filesystems 4-19

Chapter 5. LP Print Service Administration
5.1 Introduction 5-1
5.2 Starting and Stopping the LP Print Service 5-4
5.3 Printer Management 5-6
5.4 Troubleshooting 5-32
5.5 Managing the Printing Load 5-38
5.6 Managing Printer Queues 5-41
5.7 Forms Setup 5-45

Chapter 6. TTY Management
6.1 Introduction 6-1
6.2 The TTY System 6-2

Chapter 7. Basic Networking
7.1 Introduction 7-1
7.2 Networking Hardware 7-1
7.3 Networking Commands 7-2
7.4 Daemons 7-4
7.5 Supporting Database 7-5
7.6 Administrative Files 7-28
7.7 Direct Links 7-30

Appendix A. Directories and Files

Glossary

Index

Figures

2-1 Environment array for user abc 2-6
2-2 A sample */etc/profile* 2-7

4-1 The *root* filesystem 4-2
4-2 Disk partitions containing the *root* and *usr* filesystems 4-4
4-3 The relationship between disk sectors, tracks, and file blocks 4-5
4-4 Cylinders on a sample disk pack 4-6
4-5 Disk layout 4-7
4-6 Locating disk storage blocks 4-10
4-7 Sample */etc/save.d/except* file 4-17

Tables

1-1 Password Aging Character Codes 1-4

3-1 System States 3-7

About This Guide

Overview

This guide describes system administration procedures involved in the day-to-day operation of your computer system. It tells how to use the **sysadm** menu system to perform most of these tasks. For system setup, configuration, and management procedures, refer to the *System Administration Guide.*

Assumptions About the Reader

This guide is intended for system operators using the **sysadm** menu system. It assumes the reader has a basic understanding of the operating system and has some knowledge of computer operations.

Organization

This guide contains two parts: a procedures section and an information section. Chapters P1 through P7 describe procedures (typically found on **sysadm** menus) that you can use to set up and maintain your system. Chapters 1 through 7 correspond to the procedures chapters and provide more detailed information.

Notational Conventions

The following notational conventions are used in this manual:

- Filenames and user-defined parameters in text are shown in *italics.*
- System output is shown in `even-width` font.
- Commands in text and examples of input to be entered literally are shown in **bold.**
- Environment variables are shown in **bold** upper- or lowercase.
- When the name of a command or file is followed by a number in parentheses, such as **mv**(1), the number refers to the section in the *Reference Manual* that contains the page describing that command or file.

Using sysadm

This chapter introduces you to the **sysadm** menu system. It tells how to start up and exit the menu system, how to access the menu help messages, how to select menu items, and how to execute commands. As you read this chapter, it might be helpful to look at the menu system and try out some of its features.

Starting Up sysadm

To use the **sysadm** menu system, you must log in either as **root** or as **sysadm**. Then type **sysadm**, press Return, and the System Administration Menu will appear.

```
                    SYSTEM ADMINISTRATION

1 diskmgmt          disk management menu
2 filemgmt          file management menu
3 machinemgmt       machine management menu
4 packagemgmt       package management
5 softwaremgmt      software management menu
6 syssetup          system setup menu
7 ttymgmt           tty management menu
8 usermgmt          user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP, q to QUIT:
```

The System Administration Menu lists several other menus that contain subcommands for related tasks. For example, the User Management Menu contains subcommands for maintaining user information and the TTY Management Menu contains subcommands for maintaining the TTY system.

Menu Help

Help is available for each item on a menu. To see a help message, type the number corresponding to the item, followed by a question mark (?). Then press Return. For example, to see the help message for item 8, `usermgmt`, you would type **8?**.

```
Enter a number, a name, the initial part of a name, or
? or <number>? for HELP, q to QUIT: 8?
```

When you finish reading the help message, press Return to go back to the menu.

Leaving the Menu System

When you finish working in the menu system, type **q** to exit to the shell prompt.

Selecting Items from a Menu

To select an item from a menu, type either the number corresponding to the item or the first few letters of the name. For example, to select the User Management Menu, you could type either **8** or **user**.

The submenu you selected then appears on the screen. To return to your previous menu, type a caret (^).

Most menus list one or more subcommands, as shown on the User Management Menu.

```
                    USER MANAGEMENT

1 addgroup      add a group to the system
2 adduser       add a user to the system
3 delgroup      delete a group from the system
4 deluser       delete a user from the system
5 lsgroup       list groups in the system
6 lsuser        list users in the system
7 modadduser    modify defaults used by adduser
8 modgroup      make changes to a group on the system
9 moduser       menu of commands to modify a user's login

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP, ^ to GO BACK, q to QUIT:
```

You can select an item from a submenu in the same way you select items from the System Administration Menu. You can type either the number corresponding to the item or you can type the name (or the first part on the name) of the item. For example, to select the `adduser` item, you could type **2**, **adduser**, or **addu**.

Executing a Subcommand

When you select a subcommand from a menu, you start up a program that performs a task for you. Some subcommands simply print a report on your screen. For example, when you select the **lsuser** option, a list of the users on your system appears on your screen.

Other subcommands print a series of prompts on the screen. These prompts ask for the information the system needs to execute the subcommand. For example, these prompts appear when you select the `addgroup` option.

```
Enter group name [?,q]:

Enter group ID number (default 100) [?,q]:
```

At each prompt, type the appropriate response and press Return. If you make a typing error, press BackSpace to erase the character preceding the cursor.

Many prompts have been assigned a default value. To accept the default value, simply press Return.

If help is available for the prompt, a question mark follows the prompt, as shown on the previous screen. To see the help message, type **?** and press Return. If the prompt can be answered only by yes or no, the notation `y, n` follows the prompt. Type **y** or **n** as appropriate and press Return.

If you decide not to complete the procedure, you can press **q** at any prompt to terminate the procedure. Then press Return to go back to the menu from which you selected the procedure.

For many subcommands, when you finish answering the prompts, the system prints your entries on the screen. You can then edit your responses before the subcommand executes.

Bypassing the Menus

You can execute any **sysadm** subcommand directly from the command line. First type **sysadm**, then type the subcommand. For example, to execute the **adduser** subcommand (found on the User Management Menu), you could enter this command:

```
$ sysadm adduser
```

If a password has been assigned to the **sysadm** command, you will be prompted to enter the password. The **adduser** prompts then appear, just as if you selected **adduser** from the menu.

When the procedure completes, you will be returned to the shell command line.

Special Subcommands

The following **sysadm** commands can be used as logins:

- **bupsched**
- **checkfsys**
- **mountfsys**
- **powerdown**
- **setup**
- **umountfsys**

You must log in at the console; the command executes after the login procedure completes. When the command finishes running, you will be returned to the `console login:` prompt.

You can also execute these commands from the shell command line; it is not necessary to precede the command with **sysadm**.

Typically, passwords are assigned to these commands. You will be prompted for the password when you execute the command either as a login or from the command line.

Chapter P1
Identification and Security Procedures

P1.1 Set Time and Date P1-1

P1.2 Establish or Change System and Node Names P1-3
System Administration Menu—**nodename** P1-3
Command—**uname** P1-4

P1.3 Assign Passwords to Administrative Logins P1-5

Chapter P1
Identification and Security Procedures

This chapter provides the procedures you'll need to manage your system's identification and security setup. Chapter 1 in the second half of this guide contains more detailed information about identification and security issues.

This chapter includes the following procedures:

- Setting the time and date of the internal system clock
- Defining the system and node names (a node name is necessary for basic networking)
- Assigning passwords to administrative and system logins

P1.1 Set Time and Date

Purpose	To synchronize system time with clock time or to reset the system time after it has been corrupted
Starting Conditions	System state —multiuser to synchronize system time with current clock time —single-user to set the date ahead Login—**root** to use **date**(1) command to reset clock
sysadm menu	SYSTEM SETUP
Commands	**sysadm datetime**(1) **date**(1)—requires logging in as **root**
Caution	Go to single-user mode if you are setting the date ahead.

Setting the date ahead by one or more days should be done in the single-user mode. Avoid setting the date ahead while the system is in multiuser mode and **cron** is running. If you set the date while **cron** is running, it will try to catch up for the time interval involved and will start up all processes scheduled to run in the time interval.

When the operating system is booted, you might receive a reminder to set the time-of-day clock. This should happen infrequently. Setting the clock, however, is normally required when you first receive the system.

To set the time and the date, use the **datetime** option from the System Administration Menu. To accept a default value, simply press Return.

```
# sysadm datetime

Running Subcommand 'datetime' from menu 'syssetup',
SYSTEM SETUP

Current time and time zone is: 04:59 EDT
Change the time zone? [y, n, q, ?] n
Current date and time:  Tue. 08/28/85 05:00
Change the date and time: [y, n, q, ?] y
Month   default 08      (1-12): Return
Day     default 28      (1-31): Return
Year    default 84      (70-99): Return
Hour    default 05      (0-23): Return
Minute  default 00      (0-59): 04
Date and time will be set to:  08/28/84 05:04. OK? [y, n, q] y
The date and time are now changed.
```

You can also use the **date** command to reset the clock. You must be logged in as **root** to use this command and you must first take the system to single-user mode. The arguments to the **date** command are in this sequence: month, day, hour, minute, year. See **date**(1) for more information.

```
# date 0216131686
Sun Feb 16 13:16:00 EST 1986
```

P1.2 Establish or Change System and Node Names

Purpose	To define the system and node names
Starting Conditions	System state—multiuser or single-user Login—authorized login or **root**
sysadm menu	SYSTEM SETUP
Commands	**sysadm nodename**(1) **uname -S**—requires **root** login
Caution	Do not change system and node names arbitrarily. Changes must be coordinated with your networking connections.
References	Chapter 7, "Basic Networking"

There are three ways to set the system and node names:

- Use the **sysadm nodename** command.
- Use the **uname** command.
- Reconfigure the operating system after changing the SYS and NODE tunable parameters.

This section describes the first two methods. See the *System Administration Guide* for information about reconfiguring the operating system.

P1.2.1 System Administration Menu—nodename

The following screen shows an example of using the **sysadm nodename** command to set the node name. This command updates the */etc/rc.d/nodename* file, which stores the system and node names. When the system is rebooted, it reads this file to obtain the system and node names. You can use the **cat** command to read this file.

```
# sysadm nodename
Running subcommand 'nodename' from menu 'syssetup',
SYSTEM SETUP

This machine is currently called "mwa2"
Do you want to change it? [y, n, ?, q] y
What name do you want to give it? [q] ABcd5678
# cat /etc/rc.d/nodename
#         Node name changed 02/01/86 14:55:49.
uname -S ABcd5678
#
```

P1.2.2 Command—uname

On the following screen, we used the **uname** command to print and to change the system and node names. (These names are usually the same.) The name you select can be up to eight characters long.

```
# uname -a
mwa2 mwa2 3.2.0 V1.0 i386
# uname -S ABcd5678
# uname -a
ABcd5678 ABcd5678 3.2.0 V1.0 i386
#
```

The **uname -s** command does not edit the */etc/rc.d/nodename* file, which stores the system and node names. When you use **uname -s** to set the system and node names, the names will be in effect only until the next system reboot.

P1.3 Assign Passwords to Administrative Logins

Purpose	To restrict access to various administrative and special system functions
Starting Conditions	Multiuser or single-user state, any login
sysadm menu	SYSTEM SETUP
Commands	**sysadm admpasswd**(1) **sysadm syspasswd**(1)
Reference	Chapter 1, "System Security"

After setting up your computer, you should assign passwords to the special administrative and system logins (see Chapter 1, "System Security," for definitions of these logins). The **sysadm admpasswd** command allows you to assign an initial password or to change an existing password for any of the administrative logins (**setup**, **powerdown**, **sysadm**, **checkfsys**, **bupsched**, **mountfsys**, and **umountfsys**). The following screen shows an example.

```
# sysadm admpasswd

Running subcommand 'admpasswd' from menu 'syssetup',
SYSTEM SETUP
Do you want to give passwords to administrative logins? y

The login 'sysadm' already has a password.
Do you want to change the password, delete it, or skip it?
[c, d, s, q, ?] s
Password unchanged.
The login 'bupsched' already has a password.
Do you want to change the password, delete it, or skip it?
[c, d, s, q, ?] q
Password unchanged.
#
```

The **sysadm syspasswd** command allows you to assign an initial password to the **root** and **nuucp** system logins. You cannot use this command to change an existing password. When you run the **syspsswd** command, you must be logged in either as **root** or as one of the system logins.

```
# sysadm syspasswd
Running subcommand 'syspasswd' from menu 'syssetup',
SYSTEM SETUP
Do you want to give passwords to system logins? [y, n, ?, q] y
Do you want to give the 'nuucp' login a password? [y, n, ?, q] n

The following system logins still do not have passwords:
nuucp
```

Chapter P2
User Services Procedures

P2.1 Add Users or Groups P2-1

P2.2 Modify User Information P2-4
Changing Default Values for **adduser** P2-4
Changing Group Information P2-5
Changing User Information P2-6

P2.3 Delete Users or Groups P2-7

P2.4 List Users or Groups P2-9

P2.5 Broadcast Messages to All Users P2-11

Chapter P2
User Services Procedures

This chapter provides procedures you can use to maintain user and group information and to broadcast messages to users. Chapter 2 in the second half of this manual contains more detailed information about users and groups and about communication services.

This chapter includes the following procedures:

- Adding new users or groups to the system
- Changing information about existing users or groups
- Removing users or groups from the system
- Displaying information about users or groups
- Broadcasting messages to users who are logged in

P2.1 Add Users or Groups

Purpose	To identify new users or groups to the system
Starting Conditions	System state—multiuser
sysadm menu	USER MANAGEMENT
Commands	**sysadm adduser**(1) **sysadm addgroup**(1)
Reference	Chapter 2, "User Services"

To add a new user, run the **sysadm adduser** menu option. The following screen shows an example of this procedure. (Press Return to accept a default value appearing on the screen.)

```
Running subcommand 'adduser' from menu 'usermgmt',
USER MANAGEMENT

Any time you want to quit, type "q".
If you are not sure how to answer any prompt, type "?" for help.

If a default appears in the question, press <RETURN> for the default.

Enter user's full name [?, q]: John Q. Public
Enter user's login ID [?, q]: jqp
Enter user's ID number (default 46145) [?, q]: Return
Enter group ID number or group name
(default 1)[?, q]: Return
Enter user's login (home) directory name.
(default '/usr/jqp')[?, q]: Return

This is the information for the new login:
   User's name:     John Q. Public
   login ID:        jqp
   user ID:         46145
   group ID:        1
   home directory: /usr/jqp
Do you want to install, edit or skip this entry [i, e, s, q]? i
Login installed
Do you want to give the user a password? [y, n] y
New password:
(Enter at least six characters, one of them a numeral.)
Re-enter new password:

Do you want to add another login?[y, n, q]  n
```

To add a new group, use the **sysadm addgroup** command. The following screen shows an example of this procedure.

```
Running subcommand 'addgroup' from menu 'usermgmt',
USER MANAGEMENT

Any time you want to quit, type "q".
If you are not sure how to answer any prompt, type "?" for help.

If a default appears in the question, press <RETURN> for the default.

Enter group name [?, q]: seventy7
Enter group ID number (default 45201) [?, q]: Return

This is the information for the new group:
   Group name:     seventy7
   group ID:       45201
Do you want to install, edit or skip this entry [i, e, s, q]? i
Group installed

Do you want to add another group?[y, n, q]  n
```

P2.2 Modify User Information

Purpose	To change user information
Starting Conditions	System state—multiuser
sysadm menu	USER MANAGEMENT
Commands	**sysadm modadduser**(1) **sysadm modgroup** **sysadm moduser**(1) **sysadm chgloginid**(1) **sysadm chgpasswd**(1) **sysadm chgshell**(1)
Reference	Chapter 2, "User Services"

This procedure describes three functions:

- Changing the default values that apply to the **adduser** sequence (**modadduser**)
- Changing the name of a group (**modgroup**)
- Changing user information (**moduser**)

Changing Default Values for adduser

The **sysadm modadduser** command allows you to change the default values for the group ID and home (parent) directory that appear on the **adduser** form. On the following screen, we changed the default group number from **1** to **100**.

```
Running subcommand 'modadduser' from menu 'usermgmt'
USER MANAGEMENT

Any time you want to quit, type "q".
If you are not sure how to answer any prompt, type "?" for help.

Current defaults for adduser:
group ID                1       (other)
parent directory        /usr
Do you want to change the default group ID? [y, n, ?, q] y
Enter group ID number or group name [?, q] 100
Do you want to change the default parent directory? [y,n,?,q] n

These will be the new defaults:
group ID:               100
parent directory:       /usr
Do you want to keep these values? [y, n, q] y
Defaults installed.
```

Changing Group Information

To change the information for a group, use the **sysadm modgroup** command. The Modify Group Attributes Menu then appears on your screen. This menu allows you to change the name of the group.

Changing User Information

To modify the information for a user login, run the **sysadm moduser** command. The following menu then appears.

```
MODIFY USER'S LOGIN

1 chgloginid      change a user's login
2 chgpassword     change a user's password
3 chgshell        change a user's login shell

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP, q to QUIT:
```

When you select an item from this menu, you will see a prompt sequence that will help you make the required change. For example, when a user account is set up, a default shell program such as **/bin/sh** is assigned. The **chgshell** subcommand lets you assign a different shell. To use this subcommand, select **chgshell** from the menu or enter the following command at the shell prompt:

```
$ sysadm chgshell
```

The following example assigns a restricted shell (**/bin/rsh**) to user *jqp*. (See Chapter 1 for more information about restricted shell programs.)

```
Running subcommand 'chgshell' from menu 'moduser',
MODIFY USER'S LOGIN

Enter user's login ID [?, q]: jqp
The current shell is /bin/sh
Enter new shell command[q]: /bin/rsh

Do you want to change the login shell of another login?[y,n,q] q
```

P2.3 Delete Users or Groups

Purpose	To remove an inactive user from the system; to eliminate a group name that is no longer needed
Starting Conditions	System state—multiuser
sysadm menu	USER MANAGEMENT
Commands	**sysadm deluser**(1) **sysadm delgroup**(1)
Caution	When you delete a user ID, all files and directories owned by that user ID are also deleted.
Reference	Chapter 2, "User Services"

To delete a group ID, use the **sysadm delgroup** command. Following is an example of the sequence that appears on your screen:

```
Which group name do you wish to delete?[q] seventy7
Do you want to delete group name 'seventy7', group ID 45201?[y,n,?,q] y
seventy7 has been deleted
Do you want to delete any other group?[y, n, q] q
```

The **sysadm delgroup** command deletes only the specified group; it does not delete the user logins assigned to that group. The logins belonging to the group must be deleted separately by using **sysadm deluser**.

The **sysadm deluser** command deletes a user login. The user's home directory and all files in and below that directory are also deleted. Here is an example:

```
$ sysadm deluser
Password:

Running subcommand 'deluser' from menu 'usermgmt',
USER MANAGEMENT

This function COMPLETELY REMOVES THE USER, the mail file, home directory
and all files below their home directory from the machine.
Once this is done, there is no way guaranteed to get them all back.
         BE SURE THIS IS WHAT YOU WANT TO DO!

Enter login ID you wish to remove[q]: jqp
         'jqp' belongs to 'John Q. Public'
         whose home directory is /usr/jqp
Do you want to remove login ID 'jqp'?[y, n, ?, q] y

/usr/jqp and all files under it have been removed.

Enter login ID you wish to remove [q]: q
```

P2.4 List Users or Groups

Purpose	To list the users or groups known to the system
Starting Conditions	System state—multiuser
sysadm menu	USER MANAGEMENT
Commands	**sysadm lsuser**(1) **sysadm lsgroup**(1)

The two **sysadm** subcommands in this procedure enable you to look at the groups and users set up on the system. To see a list of the groups included on your system, use the following command:

```
$ sysadm lsgroup
Password:
```

Following is an example of the report this command produces.

```
Groups currently in the computer
(press <RETURN> to start printing each time you hear the bell)

group     group     logins permitted to become
name      number    members using newgrp
-----     ------    --------------------------
adm       4         root,adm,daemon
bin       2         root,bin,daemon
daemon    12        root,daemon
mail      6         root
nuucp     10        uucp
other     1
root      0         root
sys       3         root,bin,sys,adm
```

Use the following command to see a list of users set up on your system:

```
$ sysadm lsuser
Password:
```

Following is an example of the report this command produces.

```
Users currently in the computer
(press <RETURN> to start printing each time you hear the bell)

login name       user name
----------       ---------
abc              Ann Corwin
adm              0000-Admin(0000)
alice            Alice Olson
betsy            Betsy Alexander
bin              0000-Admin(0000)
bob              Bob Frelander
capp             George Capp
$
```

P2.5 Broadcast Messages to All Users

Purpose	To send urgent messages to all users logged in on the system
Starting Conditions	System state—multiuser Login—**root** to prevent users from blocking messages
Commands	**wall**(1M)
Reference	Chapter 2, "User Services"

The **wall** command allows you to communicate simultaneously with all users logged in on the system. To use this command, first become superuser, then type **wall**, press Return, and type the contents of your message. When you finish your message, type Ctrl-D.

```
# wall
System coming down in ten minutes.
Please log off. Ctrl-D
#
```

The message is sent immediately to each user's terminal and is preceded by this line:

```
Broadcast Message from ...
```

A typical use of the **wall** command is to warn users that the system is about to be shut down.

```
Broadcast Message from root:  System coming down in
ten minutes.  Please log off.
```

Chapter P3
Run Level Procedures

P3.1 Powerdown P3-1
P3.2 Shutdown to Single-User P3-3
P3.3 Return to Multiuser P3-4

Chapter P3
Run Level Procedures

This chapter contains the procedures you can use to remove power from the system and to change the system run level. Chapter 3 in the second half of this manual provides more detailed information about day-to-day operations and system run levels.

This chapter describes the following procedures:

- Removing power from the system
- Bringing the system to the single-user state to perform administrative tasks
- Returning the system to the multiuser state

P3.1 Powerdown

Purpose	To halt the system and turn off the power
Starting Conditions	Login—authorized user or root
Commands	**shutdown**(1M)—**root** login only
Reference	Chapter 3, "Processor Operations"

The system must be in firmware mode when you turn off power to the system. To take the system to firmware mode, you can use the **sysadm firmware** command or the following procedure.

Step 1. Run the **shutdown** command as follows:

```
shutdown -y -i0 -g0
```

The arguments have these meanings:

-y Assume yes answers to all questions.

-i0 Go to state 0 (firmware).

-g0 Allow grace period of 0 seconds.

Messages similar to the following then appear on the console.

```
Shutdown started.

The system will be shutdown in 0 seconds.
Please log off now.
THE SYSTEM IS BEING SHUTDOWN NOW ! ! !
Log off now or risk your files being damaged.
The system is coming down. Please wait.

The system is down.
```

Step 2. You can now turn off the power to the system.

P3.2 Shutdown to Single-User

Purpose	To perform administrative tasks that should be done when no other users are on the system
Starting Conditions	System state—multiuser Login—**root**
Commands	**shutdown**(1M)
Reference	Chapter 3, "Processor Operations"

Many tasks such as unmounting filesystems must be performed when the system is in single-user mode. Because only the console has access to the system in the single-user state, try to schedule tasks requiring single-user state for off hours.

Step 1. Log in as **root** at the console.

Step 2. Enter this command:

```
# shutdown
```

By default, **shutdown** offers prompts about the broadcast messages, provides a 60-second grace period between each message, and brings the system to the single-user state. When the system reaches the single-user state, you will see the following message:

```
INIT: SINGLE USER MODE
```

You can now proceed with your administrative tasks.

P3.3 Return to Multiuser

Purpose	To make the system available to users
Starting Conditions	System state—single-user Login—**root**
Commands	**init**(1M) **/unix** (boot program)
Reference	Chapter 3, "Processor Operations"

You can bring the system back to multiuser mode from either the single-user state or the firmware mode, or you can cause the system to immediately halt and reboot. This procedure tells how to return to multiuser state from the single-user state.

After you finish your administrative tasks, you can use the **init** command to bring the system back to the multiuser state. At the console, enter this command:

```
# init 2
```

This causes **init** to inspect */etc/inittab* and to execute entries that initialize the system to the multiuser state. The following messages appear on your screen:

```
INIT: New run level: 2

The system is coming up.  Please wait.
```

The operating system checks your filesystems and prints the current system configuration followed by this message:

```
The system is ready.

Console Login:
```

The system is now in the multiuser state and you can log in either as **root** or as a conventional user.

Chapter P4
Filesystem Administration Procedures

P4.1 Mount or Unmount Filesystems P4-2

P4.2 Maintain Filesystems P4-3
Filesystem Checking P4-3
Monitoring Disk Usage P4-5

P4.3 Filesystem Backup and Restore P4-6
Complete Backup P4-6
Incremental Backup P4-8
Back Up Specific Files P4-10
Restore Files P4-12
Backup Schedule Reminders P4-14

Chapter P4
Filesystem Administration Procedures

This chapter contains procedures you can use to maintain filesystems and to monitor disk use. Chapter 4 in the second half of this chapter contains more detailed information about filesystem maintenance.

This chapter includes the following procedures:

- Mounting and unmounting filesystems
- Checking and repairing filesystems
- Monitoring disk space use
- Backing up files
- Restoring files

NOTE

Some variation in these procedures can occur depending on the configuration of your system.

P4.1 Mount or Unmount Filesystems

Purpose	To bring a filesystem under operating system control (**mount**), or to release it so it can be removed from the system (**unmount**).
Starting Conditions	System state—multiuser to mount filesystems; single-user to unmount filesystems
sysadm menus	DISK MANAGEMENT
Commands	**sysadm mountfsys** **sysadm umountfsys**
Reference	Chapter 4, "Filesystem Administration"

This procedure includes the **sysadm mountfsys** and **sysadm umountfsys** commands, which are used to make filesystems available for use or to remove them from use. Because most standard filesystems are automatically mounted and unmounted during the startup and shutdown procedures, you seldom have to use these commands.

If you want to mount a filesystem that has already been created, select **mountfsys** from the Disk Management Menu (type **sysadm diskmgmt** to see this menu). Then answer the questions that appear on the screen. You will first be asked if you want to mount the filesystem "read-only," which prevents anyone from writing to the filesystem or changing the data it contains. You will then be asked to select the drive partition containing the filesystem. To see a list of available drive partitions, type a question mark (?).

If you want to unmount a filesystem, select **umountfsys** from the Disk Management Menu. Then select the partition containing the filesystem to be unmounted.

The **umount**(1M) command can also be used to unmount filesystems. However, you should not use this command if the filesystem was mounted with **mountfsys**.

P4.2 Maintain Filesystems

Purpose	To check and possibly repair filesystems; to check disk space usage in order to provide adequate resources to users
When Performed	Before mounting the filesystem; on a schedule appropriate for your circumstances
Starting Conditions	System state—multiuser or single-user For checking, the filesystem must NOT be mounted; for other maintenance, the filesystem must be mounted To run this procedure in single-user mode, you must execute **mount /usr**
sysadm menus	FILE MANAGEMENT DISK MANAGEMENT
Commands	**sysadm checkfsys**(1) **sysadm diskuse**(1) **sysadm fileage**(1) **sysadm filesize**(1)
Reference	Chapter 4, "Filesystem Administration"

P4.2.1 Filesystem Checking

This menu option runs the **fsck** command. For more information about **fsck**, refer to Chapter 4. For a list of **fsck** error messages and corrective actions, see the *Error Messages* manual.

Step 1. Enter this command:

```
$ sysadm checkfsys
Password:
```

Step 2:. Select the appropriate drive and press Return.

Step 3. When you are asked, specify the type of checking to be performed:

```
Select:
    check
    interactive
    automatic
[c, i, a, q, ?]:
```

- If you select `check`, the command reports any errors detected, but does not fix them.
- If you select `interactive`, the command asks you to approve or disapprove fixes to errors. (You will see a message on your screen that describes the error and asks for a **yes** or **no** response.)
- If you select `automatic`, the command automatically fixes errors.

Following your response, file checking proceeds. When the check is complete, a message like this appears:

```
27 files 94 blocks 1266 free
```

P4.2.2 Monitoring Disk Usage

The second part of the filesystem maintenance procedure involves making sure that enough space is available on disk to accommodate user needs.

Step 1. Enter this command:

```
$ sysadm diskuse
Password:
```

A display similar to the following then appears on your screen:

```
FILE SYSTEM USAGE AS OF 05/04/86  13:23:30
File      Free      Total       Percent
System    Blocks    Blocks      Full
------    ------    ------      -------
/          14596     49896      70%
/usr      248342    351052      29%
```

If you want more detailed information about your filesystems, you can use the **fileage** and **filesize** commands.

Step 2:. Type the **fileage** command as follows:

```
$ sysadm fileage
```

This command prompts you for two pieces of information:

- The full pathname of the directory to be searched. It is important to be specific in your response. If you select a high-level directory such as */usr*, you will receive a great deal more information than you want.
- The number of days to go back; files older than the number of days you specify will be reported. The default is 90 days.

Step 3. Type the **filesize** command:

```
$ sysadm filesize
```

This command locates information about the *n* largest files (default is 10) in the directory you specify. The information includes the owner of the file, the number of characters in the file, the date it was last accessed, and its name.

P4.3 Filesystem Backup and Restore

Purpose	To store filesystems or parts of filesystems in order to guard against loss of data and to free up space on the disk
When Performed	On a schedule developed to fit the needs of your system
Starting Conditions	System state—single-user You must execute **mount /usr** to run this procedure in single-user mode. Login—**root**
sysadm menu	FILE MANAGEMENT
Commands	**sysadm backup**(1) **sysadm store**(1) **sysadm restore**(1) **sysadm bupsched**(1) **mount**(1M)
Media	Tapes in enough quantity to hold the files or filesystems you are backing up.
Reference	Chapter 4, "Filesystem Administration"

P4.3.1 Complete Backup

Step 1. Log in as **root**.

Step 2. Take the system to the single-user mode (run-level S or 1); see Procedure P3.2, "Shutdown to Single-User."

Step 3. Use **mountall** to mount all of your filesystems.

Step 4. Enter this command:

```
# sysadm backup
```

Because you logged in as **root**, you are not prompted for the **sysadm** password. Next, answer the questions that appear on the screen:

```
Enter file system(s) you want to backup [q, ?]: /usr2
Select complete or incremental backup [c, i, q, ?]: c
Print each file name as it is copied? [q,?] [y, n, q, ?]: y

Select which drive to use:

Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT:

Before inserting the first part into the drive, mark it:

        Complete Backup of /usr2,
        Sat. 09/08/84, 05:08:31 AM
                Part 1

Insert the medium in the drive.  Press <RETURN> when ready. [q]
(If you asked for filenames to print, they will appear here.)
Reached end of medium on output.

Remove medium.

Before inserting the next part into the drive, mark it:

        Complete Backup of /usr2,
        Sat. 09/08/84, 05:28:31 AM
                Part 2

Insert the medium in the drive.  Press <RETURN> when ready. [q]
(If you asked for filenames to print, they will appear here.)
Reached end of medium on output.
    .
(Process continues until copying is complete)
13089 blocks

Complete backup of  /usr2  has finished.
You may now remove the medium.
#
```

Step 5. Label each tape used for the backup. Include a sequence number as part of the label (such as Part 1, Part 2, and so on).

Step 6. Return the system to the normal operating configuration (see Procedure P3.3, "Return to Multiuser").

P4.3.2 Incremental Backup

Incremental backups can be used after an initial full backup has been done. See the discussion on backup strategies in Chapter 4.

Step 1. Log in as **root**.

Step 2. Take the system to the single-user mode (run-level S or 1).

Step 3. Use **mountall** to mount all of your filesystems.

Step 4. Execute the System Administration backup command (**sysadm backup**) and follow the instructions that appear on your screen.

Step 5. Label the medium as indicated in the instructions.

Step 6. Return the system to the normal operating configuration (see Procedure P3.3, "Return to Multiuser").

```
# sysadm backup
Running subcommand 'backup' from menu 'filemgmt',
FILE MANAGEMENT

Available file systems:
/ /usr /usr2 ALL
Enter file system(s) you want to backup [q, ?]: /usr2
Select complete or incremental backup [c, i, q, ?]: i
Print each file name as it is copied? [q, ?] [y, n, q, ?]: y

Select which drive to use:

Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT:

Before inserting the tape into the drive, mark it as follows:

        Incremental Backup of /usr2,
        Sat. 09/08/84, 05:08:31 AM to
        Sat. 09/08/84, 07:04:21 AM
                Part 1
Insert the medium in the drive.  Press <RETURN> when ready. [q]
/usr2/abc/fil01
   .
   .
   .
83 blocks

Incremental backup of  /usr2  has finished.
You may now remove the medium.
#
```

P4.3.3 Back Up Specific Files

Step 1. Log in as **root**.

Step 2. Take the system to the single-user mode, run-level S or 1 (see Procedure P3.2).

Step 3. Use **mountall** to mount all of your filesystems.

Step 4. Enter this command:

```
# sysadm store
```

Select the appropriate device, then follow the prompts that appear on your screen.

```
Running subcommand 'store' from menu 'filemgmt',
FILE MANAGEMENT

  1. Select a single file for storing
  2. Select all files under a directory for storing
Enter a number: 1
Enter full path name of file to be stored [q]: /usr/abc/file1

  1. Select a single file for storing
  2. Select all files under a directory for storing
  3. List files selected so far
  4. Store selected files
Enter a number: 4

  1 files selected
```

Step 5. Each time you provide a file or directory name, the system prompts you either to enter more names, to review what has been entered, or to proceed with the storing process. When you have specified all files to be stored, enter **4**. You will see a display similar to the following.

```
Files stored on:
Sat 05/04/85, 03:18:53 PM
     part 1
/usr/abc/file1
28 blocks
   .
   .
   .
Store complete.
Do you want to verify that your file(s) were stored properly[y,n,q,?] y
PLEASE NOTE:
To verify that the store worked properly, you must
re-insert all parts that were just written to, starting with "part 1"
Insert the medium.  Press <RETURN> when ready.
Verification complete.  You may remove the medium.
Should the stored files be removed from the disk[y,n,q,?]
```

The last question gives you a chance to delete the files you backed up. You would probably elect to do that if you were in the process of freeing up storage space.

Step 6. Return the system to the normal operating configuration.

P4.3.4 Restore Files

Step 1. Log in as **root**.

Step 2. Take the system to the single-user mode (run-level S or 1).

Step 3. Use **mountall** to mount all of your filesystems.

Step 4. Enter this command:

```
$ sysadm restore
```

Follow the instructions that appear on your screen. (The following page shows an example.)

When restoring from a complete or incremental backup, all tapes of that series must be loaded. Even if you intend to restore only a single file, all tapes of the backup series must be loaded in sequence.

Step 5. Return the system to the normal operating configuration (see Procedure P3.3, "Return to Multiuser").

Step 6. Store the backup tapes in a safe place.

```
# sysadm restore

Running subcommand 'restore' from menu 'filemgmt',
FILE MANAGEMENT

Select which drive to use

Enter a number, a name, the initial part of a name, or
? for HELP, q to QUIT: 2

Select:
        1. restore a single file
        2. restore a directory of files
        3. restore all files
        4. list all the files
Enter a number [q,?]: 1
Insert the medium in the drive. Press <RETURN> when ready.
Enter full path name of file(s) to be restored [q, ?]: /usr/abc/file1
Do you want to rename the file as it is copied in? [y, n, q]: y
WARNING:
Be very careful when you rename a file.  Files incorrectly named
by typing errors are difficult to find and repair.
Remember that only the first 14 characters of each part of the
file name (i.e. the characters between the "/"s) are significant.

(You will be asked to rename each file in turn. Press Return to skip a file.
Type a dot (.) to restore a file with its original name.)

Rename </usr/abc/file1>
/usr/abc/datafile3
   .
   .
   .
83 blocks

Restoration complete.
```

P4.3.5 Backup Schedule Reminders

You can use any authorized login to carry out this procedure.

Step 1. Enter this command:

```
$ sysadm bupsched
Password:
```

The following menu appears on your terminal.

```
                    BACKUP REMINDER SCHEDULING

  1 schedcheck  schedule backup reminder checks
  2 schedmsg    schedule backup reminder message

  Enter a number, a name, the initial part of a name, or
  ? or <number>? for HELP, q for QUIT:
```

Step 2. To schedule a reminder message to be sent to the system console during shutdown, enter **2**. The following message appears on your terminal.

```
Enter the command you wish to execute [p, r, a, m, w, q, ?]:
```

Type **?** to see your choices for this prompt:

```
p - Print lines of the file
r - Remove a line or group of lines
a - Add a line
m - Modify a line
w - Write the changes into the file
q - Quit - Leave bupsched
```

The file referred to contains reminder messages and the schedule for sending the messages to the system console.

Step 3. Enter **a** to add a line. The following sequence of prompts occurs:

```
Enter time intervals in which backup reminder messages are to be printed
[q, ?]: 16:00-18:00
```

(The time interval must be entered with no blanks.)

```
Enter time intervals in which backup reminder messages are to be printed
[q, ?]: q
```

(Many prompts in this sequence are repeated to let you enter additional information. When you are ready to go to the next prompt, press **q**.)

```
Enter the day of the week [0-6, *, q, ?]: *
```

(The asterisk (*) means you want to schedule the message to appear every day. Day 0 is Sunday.)

```
Enter the day of the month [1-31, *, q, ?]: 15
```

(The **15** *means you want to schedule the message to appear on the 15th of the month. This modifies the Day-of-the-Week schedule to mean any day that is the 15th of the month.)*

```
Enter the day of the month [1-31, *, q, ?]: q
```

(The prompt is repeated to let you enter additional days for this message to be scheduled. Type **q** *to end the sequence.)*

```
Enter  the month [1-12, *, q, ?]: *
```

(The asterisk (*) means you want to schedule the message to appear every month.)

```
.
.
Enter the file system you wish to backup [q, ?]: /usr2
```

(The name must start with a slash (/). The prompt will re-appear until you enter **q**.)

```
Enter the command you wish to execute [p, r, a, m, w, q, ?]: w
```

(The **w** *tells the system to add your entry to the file of reminder messages.)*

```
Enter the command you wish to execute [p, r, a, m, w, q, ?]:
```

The sequence is over. You can exit with a **q** or choose another function.

Step 4. To see all scheduled reminder messages, type **p**. A display similar to the following then appears on your terminal. A pound sign (#) at the beginning of a line means that line is interpreted as a comment when the file is executed.

```
#
#       Format of lines
#
#time       day       month       list
#
#       time   - time(s) of day (24hr or am/pm)
#       day    - day(s) of week (mon, tue, etc)
#                day(s) of month (1,2,...first, last)
#       month  - month(s) of the year (jan, feb,...)
#       list   - list of file systems to be backed up
#                or command to be executed ( !command line)
#
#       Example:
#
#16:00-18:00  mon  *  /usr
#
#       If ckbupscd is invoked between 4:00 and 6:00 in the
#       evening on mondays during any month of the year,
#       display /usr as the name of a file system that needs
#       to be backed up.
#
#================================================
#
#       Default backup schedule calls for daily backups of /usr
#       and monthly backups of root (/) on the 15th of each month.
#
   1  0:00-8:00,16:00-23.59 mon,tue,wed,thu,fri  *  /usr
   2  4pm-11pm              15       *        /
   3  16:00-18:00           15       *        /usr2
```

(Line 3 in the list is the line you just added.)

Step 5. Once there are reminder messages in the file, you can use **schedcheck** (selection 1 from the **bupsched** menu) to schedule checks for reminder messages. Reminder messages are sent to the console if **shutdown** occurs during the time interval specified in the file.

Schedule check is an added protection in case a **shutdown** does not take place during the specified time interval. It looks for messages that would have been sent if a **shutdown** had occurred. The prompt sequence is similar to the one shown earlier.

Chapter P5
LP Print Service Procedures

P5.1 Stop the Print Service P5-1

P5.2 Restart the Print Service P5-2

P5.3 Set Up the Print Service P5-3
Add a Printer P5-3
Change a Printer Configuration P5-6
Delete a Printer P5-10

P5.4 Set Up Forms P5-11
Using Forms P5-11
Add a Form P5-13
Change a Form P5-15
Delete a Form P5-17

P5.5 Set Up Filters P5-18
Using Filters P5-18
Add a Filter P5-20
Change a Filter P5-21
Delete a Filter P5-22

Chapter P5
LP Print Service Procedures

This chapter provides procedures you can use to manage the LP print service. Chapter 5 in the second half of this guide contains more detailed information about the print service.

This chapter includes the following procedures:

- Stopping the LP print service
- Restarting the LP print service
- Adding or deleting a printer
- Changing a printer configuration
- Adding, changing, or deleting a form
- Adding, changing, or deleting a filter

P5.1 Stop the Print Service

Purpose	To stop the print service
Starting Conditions	Multiuser or single-user state (If the system is in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**
Commands	**sysadm packagemgmt/lpmgmt/service/stop**
Reference	Chapter 5, "LP Print Service Administration"

Enter the following command to stop the print service:

```
$ sysadm stop
```

This command stops the print service completely. A screen message appears when the print service has stopped; no further requests for printing will be processed.

P5.2 Restart the Print Service

Purpose	To make the LP system available again
Starting Conditions	Multiuser or single-user state (If the system in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**
Commands	**sysadm packagemgmt/lpmgmt/service/start**
Reference	Chapter 5, "LP Print Service Administration"

When you are ready to restart the print service, enter the following command:

```
$ sysadm start
```

A message appears on the screen when the print service is available again.

P5.3 Set Up the Print Service

This section tells how to add a printer to the print service, how to change a printer configuration, and how to delete a printer from the print service. Before you can carry out these procedures, the printer must be connected to an available port.

You should also make sure the system can communicate with the printer. As a test, use the **cat** command to send a file to the printer. Although the file might not print properly, this step confirms that your system and the printer can communicate.

Purpose	To add or delete a printer from your system or to change a printer's configuration
Starting Conditions	Multiuser or single-user state (If the system is in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**
Commands	**sysadm packagemgmt/lpmgmt/printers/add_p** **sysadm packagemgmt/lpmgmt/printers/change_p** **sysadm packagemgmt/lpmgmt/printers/accept_p** **sysadm packagemgmt/lpmgmt/printers/enable_p** **sysadm packagemgmt/lpmgmt/service/default** **sysadm packagemgmt/lpmgmt/printers/reject_p** **sysadm packagemgmt/lpmgmt/printers/disable_p** **sysadm packagemgmt/lpmgmt/printers/delete_p**
Reference	Chapter 5, "LP Print Service Administration"

P5.3.1 Add a Printer

To add a printer to the LP print service, complete the steps outlined in this section.

Step 1. To specify the printer's configuration, enter the following command:

```
$ sysadm printers/add_p
```

If other printers have already been added to the system, you will be asked if the printer you are now adding is similar to any of them. If it is, the configuration for the similar printer can be used as a template for configuring the

new printer. If there is not a similar printer, the system will supply standard defaults for the printer configuration. You can accept or change any of the default values. A sample printer configuration follows.

(If you define the alert-type for the printer to be **mail** or **write** and do not specify a login, the login **sysadm** will be assumed.)

```
Enter the name of the new printer:  printer2
Is this printer to be similar to one already added? [y, n, ?, q] n
Printer class to add it to: (default: none)  letterquality
Enter one line that describes the printer for your users.
(default: none)  daisy-wheel printer
Type of printer: (default: unknown)  455
Types of files printable without filtering:
(default: simple) simple,nroff

Printer interface to use: (default: standard)  /usr/kol/interface
Can a user skip the banner page? (default: no) [y, n, ?, q]

Use the default page size and printing pitch? (default: yes) [y, n, ?, q]

Enter a command to run to alert you when the printer faults.
(default: mail lp)  write lp
How often should you be alerted (minutes, 0=once)? (default: 0)  3
How should printing restart after you fix a printer fault?
(default: continue)

Is the printer directly connected? (default: yes) [y, n, ?, q]
Printer port to use:  /dev/ttyGA/GAAH
Is the printer also a login terminal? (default: no) [y, n, ?, q]

The default port settings will be:
        9600 cs8
    cstopb -parenb -parodd
    ixon -ixany
    opost -olcuc onlcr -ocrnl -onocr -onlret
    -ofill nl0 cr0 tab0 bs0 vt0 ff0
Enter any changes or additional settings that must be in
effect:

List the print wheels available:
(default: none) elite, courier
```

```
List the users who are denied access to this printer.
(default: none)
List the users who are allowed access to this printer.
(default: all)  merlyn ehr3 glg

List the forms that can be used with this printer.
(default: none)  payroll_check, order_form

install, edit, or skip this printer entry: (default: install)
```

If you have trouble defining a value for a field, be sure to read the help messages and look for further details in Chapter 5 of this guide.

When you finish this session, the printer you defined will be added to the LP print service. Next, you must tell the print service that the printer can accept print requests into its queue. You must also enable the printer so that it can print the requests.

Step 2. To tell the print service to accept print requests for the printer you added, enter the following command:

```
$ sysadm printers/accept_p
```

You will be asked to list the printers or printer classes that can begin accepting print requests.

Step 3. When you are ready to start printing, be sure that the printer is ready to receive output. For some printers, this means the top of the form has been adjusted and that the printer is online. To allow printing to occur on the printer, enter the following command:

```
$ sysadm printers/enable_p
```

You will be asked to list the printers that can begin printing requests.

Step 4. (This step is optional.) Set the default destination for print requests by entering the following command:

```
$ sysadm default
```

You will be asked to name the printer or printer class that is to be used as the default destination for print requests.

Step 5. Enter the following command to verify that the new printer has been added according to your specifications and that its configuration is correct.

```
$ sysadm printers/list_p
```

This command prompts you for the names of printers for which you want to see a configuration listing. If you specify **all**, you will receive a configuration listing of all available printers, including the one you just added.

P5.3.2 Change a Printer Configuration

When you change a printer configuration, keep in mind that print requests sent to that printer can also be affected.

To safeguard existing print requests, consider taking either or both of the following steps before you change the configuration of a printer or remove it from the print service:

- Tell the printer to reject print requests.
- Disable the printer.

When you are satisfied that no jobs are in danger of being ruined and no new jobs will be queued for the affected printer, you can delete the printer or change its configuration. The procedures for these precautionary measures are given in Steps 1 and 2.

Step 1. (This step is optional.) To tell a printer to stop accepting print requests, enter the following command:

```
$ sysadm printers/reject_p
```

You will be asked to specify the printers or printer classes that are to reject print requests. You can also specify the reason the printer is not accepting print requests.

Step 2. (This step is optional.) To disable a printer, enter the following command:

```
$ sysadm printers/disable_p
```

You can preserve or delete the current print request before the specified printer is disabled. The following screen shows an example.

```
Enter the names of the printers that should stop printing requests.
Separate multiple names with a space or comma.
(default: printer2)
Cancel any requests currently printing? (default: no) [y, n, ?, q]
Wait for any requests currently printing? (default: no) [y, n, ?, q] yes
Reason for disabling:
(default: none)  to reconfigure printer2
```

The printer specified as the default on this screen is `printer2` because it was the last printer added to the system with the **add** command. (This printer is the default for this menu option; this does not mean it is the default system printer.) If no printers have been added in this way, the default value will be `none`.

Step 3. This step tells how to change the configuration of printer. Begin by entering the following command:

```
$ sysadm printers/change_p
```

You will then be asked to specify the printer and to make any configuration changes. The screen on the following page shows a example sequence. The steps are the same as the steps you followed to add a new printer. (If you define the alert-type for the printer to be **mail** or **write** and do not specify a login, the login **sysadm** will be assumed. You cannot specify the alert-type **quiet**.)

```
Enter the name of the printer: (default: printer2)
Printer class to add it to: (default: letterquality)
Enter one line that describes the printer for your users.
(default: AT&T Model 455 daisy-wheel printer)
Type of printer: (default: 455)
Types of files printable without filtering: (default: simple, nroff)

Printer interface to use: (default: /usr/kol/interface)
Can a user skip the banner page? (default: no) [y, n, ?, q]

Use default page size and printing pitch? (default: yes) [y, n, ?, q]

Enter a command to run to alert you when the printer faults.
(default: write lp)
How often should you be alerted (minutes, 0=once)? (default: 3)  10
How should printing restart after you fix a printer fault?
(default: continue)

Is the printer directly connected? (default: yes) [y, n, ?, q]
Printer port to use: (default: /dev/ttyGA/GAAH)
Is the printer also a login terminal? (default: no) [y, n, ?, q]

The default port settings will be:
        9600
cs8 cstopb -parenb -parodd
ixon -ixany
opost -olcuc onlcr -ocrnl -onocr -onlret
-ofill nl0 cr0 tab0 bs0 vt0 ff0
Enter any changes or additional settings that must be in
effect: (default: as listed above)

List the print wheels available:
(default: elite, courier)

List the users who are denied access to this printer.
(default: none)
List the users who are allowed access to this printer.
(default: merlyn ehr3 glg) jwoo

List the forms that can be used with this printer.
(default: payroll_check, order_form)

install, edit, or skip this printer entry:  (default:  install)
```

Step 4. To tell the printer to accept print requests, enter this command:

```
$ sysadm printers/accept_p
```

Step 5. The printer must be reenabled to allow it to print. Enter the following command:

```
$ sysadm printers/enable_p
```

You will be asked for the names of printers that can start printing requests.

Step 6. (This step is optional.) To define a default destination for print requests, enter the following command:

```
$ sysadm default
```

You will be prompted for the name of the printer or printer class that is to be used as the default destination for print requests.

Step 7. The reconfiguration of your printer is now complete. If you want to verify that the configuration is correct, enter the following command:

```
$ sysadm printers/list_p
```

This command prompts you for the names of printers for which you want to see a configuration listing. If you specify **all**, you will receive a configuration listing of all available printers, including the one you have just changed.

P5.3.3 Delete a Printer

Before deleting a printer from the print service, you should either tell the printer to reject print requests or disable it. These procedures are described in the previous section. When you are ready to delete the printer, enter the following command:

```
$ sysadm printers/delete_p
```

You will be asked to name the printer to be deleted. After you specify a name, the message `Are you sure?` appears. This message allows you to make sure you are removing the correct printer.

To verify that the appropriate printer has been deleted, enter the following command:

```
$ sysadm printers/list_p
```

This command prompts you for the names of printers for which you want to see a configuration listing. If you specify **all**, the configuration should no longer include the printer you deleted.

P5.4 Set Up Forms

Purpose	To add, change or delete a form for the print service
Starting Conditions	Multiuser or single-user state (If you are in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**
Commands	**sysadm packagemgmt/lpmgmt/forms/add_f** **sysadm packagemgmt/lpmgmt/forms/change_f** **sysadm packagemgmt/lpmgmt/forms/delete_f** **sysadm packagemgmt/lpmgmt/forms/mount_f** **sysadm packagemgmt/lpmgmt/forms/list_f**
Reference	Chapter 5, "LP Print Service Administration"

P5.4.1 Using Forms

Many printers can print on a variety of pre-printed forms such as checks and invoices. The LP print service can help you take advantage of this capability, but you must supply additional software, as well as the forms. Once you have all of the necessary components in place, the LP print service can help you use your forms and the supporting software in the following ways:

- The print service can schedule the work of one or more printers, allowing you to print specific forms on specific printers at certain time periods. (If you have only one printer, the scheduling service allows you to allocate its time among multiple forms.)
- The print service can keep track of forms currently mounted and can alert you to mount forms as needed.

The steps involved in setting up a system and using it to produce pre-printed forms listed on the following page. The steps lead you through a sample case, showing the actions taken by a company that wants to automate printing payroll checks. (The database containing the payroll information is assumed to exist on the company's computer.)

1. The company buys (or asks a programmer to write) application software (a program) that takes information from the payroll database to print a standard check.
2. The system administrator installs the application on the system.
3. The system administrator notifies users (in this case, a company clerk responsible for preparing payroll checks) that the application program is available to prepare the checks.
4. The system administrator uses the **sysadm** menu for LP forms to identify the payroll checks to the print service. The LP print service provides this step.
5. The payroll clerk issues a request to the print service to print checks.
6. The print service notifies the administrator of the payroll clerk's request and alerts the administrator to change the paper in the printer from the current stock to checks.
7. The system administrator disables the printer.
8. The system administrator loads checks into the printer.
9. The system administrator enables the printer.
10. The system administrator notifies the LP print service that the checks are loaded.
11. The printer prints the payroll checks.
12. The system administrator disables the printer and loads the normal printer paper.
13. The system administrator enables the printer.

The LP print service does not provide any application software (a program that converts a file to a format suitable for a particular form, such as an invoice). You can purchase application software separately or you can write your own applications.

P5.4.2 Add a Form

Step 1. To add a form to the print service, enter the following command:

```
$ sysadm forms/add_f
```

You will be asked to name the form and to provide specifications for it, as shown in the following example.

```
Name of form:  payroll
Page length: (default: 66)
Page width: (default: 80)
Lines per inch: (default: 6)
Characters per inch: (default: 10)

Number of pages in the form: (default: 1)  2
Character set to use with the form: (default: any)
Ribbon color to use with the form: (default: any)  black
Enter a one-line description of this form for your users:
payroll check for non-supervisory staff
Enter the full pathname of a file containing an alignment pattern:
(default: none) /usr/sam/align.check
What type of file is this? (default: simple)

Enter a command to run to alert you when the form needs mounting.
(default: mail lp)  write lp
How many print requests should be waiting before you are alerted?
(default: 1)  4
How often should you be alerted (minutes, 0=once)?
(default: 0)  3

List the users who are denied access to this form.
(default: none)
List the only users who are allowed access to this form.
(default: all)  merlyn ehr3 glg

install, edit, or skip this form:   (default: install)
```

To verify that the new form has been added, enter the following command:

```
$ sysadm forms/list_f
```

This command prompts you to name the forms for which you want to see a complete description. If you specify **all**, you will receive a list of all available forms, including the one you just added.

Step 2. (This step is optional.) If you would like to mount this form (load it on the printer), enter the following command:

```
$ sysadm forms/mount_f
```

You will be prompted to answer the following questions:

```
Enter the name of the printer: (default: printer2)  printer3
Enter the name of the form you are mounting: (default: payroll)
Enter the name of the print wheel you are mounting: (default: none)
Print an alignment pattern?  [y, n, ?, q]  yes

Press return to print an alignment pattern [q to quit]:
```

The third prompt (`Enter the name of the print wheel you are mounting:`) appears only if the printer uses print wheels. If you are prompted for the names of both a form and a print wheel, be sure that you assign a value other than `none` to one of them.

The default form is **payroll** because that was the last form added to the system with the **add** command from the **forms** menu. If the form includes an alignment pattern, the prompt `Press return to print an alignment pattern [q to quit]:` appears. When you press Return, an alignment pattern prints on the specified printer. This prompt reappears until you enter **q** to stop printing the alignment pattern.

NOTE

You can also use the Printer Management Menu to mount a form. The command line **sysadm mount_p** *is equivalent to the mount command shown earlier.*

P5.4.3 Change a Form

To change the specifications for a form, enter the following command:

```
$ sysadm forms/change_f
```

The following screen shows a sample sequence. The last form added to the print service is listed as the default form; the default values for each prompt are taken from this form. You can change any of these values.

When you complete your changes, you can use the following command to check the new configuration.

```
$ sysadm forms/list_f
```

This command prompts you for the names of forms for which you want to see a complete description. To see descriptions of all available forms, including the one you have just changed, specify **all**.

```
Name of form: (default: payroll)
Page length: (default: 66)
Page width: (default: 80)
Lines per inch: (default: 6)
Characters per inch: (default: 10)

Number of pages in the form: (default: 2)
Character set to use with the form: (default: any)
Ribbon color to use with the form: (default: black)
Enter a one-line description of this form for your users:
(default: payroll check for non-supervisory staff)

Do you want to see the current alignment pattern? (default: no)

Do you want to keep the current alignment pattern? (default: yes) n

Enter the full pathname of a file containing an alignment pattern:
(default: /usr/sam/align.check) /usr/joan/align.new
What type of file is this? (default: simple)

Enter a command to run to alert you when the form needs mounting.
(default: write lp)  mail lp
How many print requests should be waiting before you are alerted?
(default: 4)  10
How often should you be alerted (minutes)?
(default: 3)

List the users who are denied access to this form.
(default: none)
List the users who are allowed access to this form
(default: merlyn, ehr3, glg)

install, edit, or skip this form:  (default:  install)
```

P5.4.4 Delete a Form

To delete a form from the print service, enter the following command:

```
$ sysadm forms/delete_f
```

You will be asked to name the forms you want to remove. Before the system deletes the forms, the message `Are you sure?` appears, allowing you to confirm that you are deleting the correct form.

Step 2. If you want to verify that the form was deleted, enter the following command:

```
$ sysadm forms/list_f
```

This command prompts you for the names of forms for which you want to see a complete description. Specify **all** to see descriptions of all available forms.

P5.5 Set Up Filters

Purpose	To add, change, or delete a filter from the print service
Starting Conditions	Multiuser or single-user state (If the system is in single-user state, you must run the command **mount /usr** before following this procedure.) Login—**root**
Commands	**sysadm packagemgmt/lpmgmt/filters/add_f** **sysadm packagemgmt/lpmgmt/filters/change_f** **sysadm packagemgmt/lpmgmt/filters/delete_f** **sysadm packagemgmt/lpmgmt/filters/list_f**
Reference	Chapter 5, "LP Print Service Administration"

P5.5.1 Using Filters

When the LP print service prints a file, it can transform the data from the format used in the file to another format. The device the print service uses to transform formats is a program called a *filter*. You must provide any filters you want to use on your system (either by buying or writing them); the LP print service does not provide filters.

The print service helps you manage filters that you have installed on your system. Specifically, it oversees the use of filters, determining when they are needed for print jobs and then matching the appropriate filter with a file and a printer.

If you are not familiar with filters, review the information in Chapter 5 before you use the procedures described in this section.

The following list shows the steps involved in using filters with the LP print service.

1. A company identifies the filters it needs, based on the types of files it typically prints.
2. The company buys or asks programmers to write those filters.
3. The system administrator adds the filters to the filter table.
4. Users issue print requests, specifying the type of file and the printer.
5. The LP print service selects a filter.
6. The filter transforms the data into the desired format.
7. The printer prints the document.

P5.5.2 Add a Filter

To add a filter to the print service, enter the following command:

```
$ sysadm filters/add_f
```

The system then prompts you to enter the name of the filter and its specifications. Following is an example of this procedure:

```
Filter name: 450
Input types it can convert: (default: any)  nroff
Output types it can produce: (default: any )  450
Printer types it is restricted to: (default: any) 455
Printers it is restricted to: (default: any)
Is this a slow filter? (default: yes) [y, n, ?, q]
Enter the filter command and any fixed options:
450
      .
      .
      .
Keyword:  MODES
Pattern:  landscape
Template:  -l

Keyword:  done

install, edit, or skip this filter: (default: install)
```

To verify that the new filter has been added, enter the following command:

```
$ sysadm filters/list_f
```

The system then prompts you for a list of the filters for which you want to see a description. Specify **all** to see descriptions of all available filters, including the one you just added.

P5.5.3 Change a Filter

To change the specifications for a filter, enter the following command:

```
$ sysadm filters/change_f
```

The system then prompts you for the name of the filter you want to change.

If the filter was delivered with the LP print service, you will be asked if you want to restore the filter to the `factory setting`. If you answer no, the system will prompt you for a new specification for the filter.

In the following example, we changed the specification for the filter that we added in the previous section. (Because we are not dealing with a filter that was delivered with the LP print service, we are not given the choice of restoring the filter to the `factory setting`.) We will make two changes: we will restrict the filter to another type of printer and we will specify that this filter is not a "slow" filter.

```
Filter name (default: 450)
Input types it can convert: (default: nroff)
Types it can produce: (default: 450)
Types of printers it is restricted to: (default: 455) 455, 480
Printers it is restricted to: (default: any)
Is this a slow filter? (default: yes) [y, n, ?, q]  no
Enter the filter command and any fixed options:
(default: 450)
      .
      .
      .
Keyword: (default: MODES)
Pattern: (default: landscape)
Template: (default: -l)

Keyword: done

install, edit, or skip this filter: (default: install)
```

To verify that the appropriate filter has been changed, enter the following command:

```
$ sysadm filters/list_f
```

This command prompts you for the names of filters for which you want to see a complete description. Specify **all** to see a list of all available filters, including the one you just changed.

P5.5.4 Delete a Filter

To delete a filter from the print service, enter the following command:

```
$ sysadm filters/delete_f
```

The system prompts you for the name of the filter you want to remove. The message `Are you sure?` then appears, allowing you to confirm your request.

To verify that the appropriate filter has been deleted, enter the following command:

```
$ sysadm filters/list_f
```

This command prompts you for the names of filters for which you want to see a complete description. Specify **all** to see a list of all available filters.

Chapter P6
TTY Management Procedures

P6.1 Check TTY Line Settings P6-1
P6.2 Create TTY Line Settings P6-4
P6.3 Modify TTY Line Characteristics P6-6

Chapter P6
TTY Management Procedures

This chapter contains procedures you can use to maintain the TTY subsystem. Chapter 6 in the second half of this manual contains more detailed information about the TTY subsystem.

This chapter describes the following procedures:

- Viewing existing TTY line settings
- Creating new TTY line settings and hunt sequences
- Changing characteristics of TTY lines

P6.1 Check TTY Line Settings

Purpose	To view existing line settings
Starting Conditions	System state—multiuser or single-user Login—an authorized login
sysadm menu	TTY MANAGEMENT
Commands	**sysadm lineset**(1)
Reference	Chapter 6, "TTY Management"

Enter this command to go directly to the lineset display:

```
$ sysadm lineset
Password:
```

A display similar to the following then appears on your terminal.

```
Running subcommand 'lineset' from menu 'ttymgmt',
TTY MANAGEMENT
Tty Line Settings and Sequences

console1   console2   console3   console4   console5   console
contty1    contty2    contty3    contty4    contty5    contty
contty1H   contty2H   contty3H   contty4H   contty5H   contty6H
pty        (does not sequence)
300        19200      9600       4800       2400       1200
300H       4800H      9600H      19200H     2400H      1200H
```

Each line setting is a name used to identify a set of TTY line characteristics. During the **login** process, the line settings within a row "hunt" from left to right, moving from one setting to the next upon receiving a BREAK signal. The rightmost setting in each row hunts to the first setting in the row, forming a circular hunt sequence.

Note that the `pty` setting does not sequence. Sending a BREAK has no effect.

The next portion of the display allows you to look at a line setting in detail. You can select any setting; you don't have to start with the first setting in a row.

```
Select one line setting to see it in detail [?, q]: 1200H

Line Setting:                1200H
       Initial Flags:        B1200
       Final Flags:          B1200 SANE IXANY TAB3 HUPCL
       Login Prompt:         login:
       Next Setting:         300H

B1200   1200 Baud
HUPCL   Hang Up on Last Close
IXANY   Enable Any Character to Restart Output
SANE    Set All Modes To "Traditionally Reasonable" Values
TAB3    Expand Horizontal-tab To Spaces

Select another line setting or
<RETURN> to see the original list [?, q]:
```

P6.2 Create TTY Line Settings

Purpose	To create new line settings and hunt sequences
Starting Conditions	System state—multiuser or single-user Login—an authorized login
sysadm menu	TTY MANAGEMENT
Commands	**sysadm mklineset**(1)
Reference	Chapter 6, "TTY Management"

In this procedure, we are connecting a dual-speed modem that can handle 300 baud and 1200 baud. There is a 1200 setting already in the table, but if users somehow miss the speed, they will have to press Break several times to hunt to 1200 again. Since we are interested in only the two speeds, we'll create a new 1200-300 sequence.

Enter this command to go directly to the **mklineset** display:

```
$ sysadm mklineset
Password:
```

The sequence of prompts shown on the next page then appears on your terminal.

```
Running subcommand 'mklineset' from menu 'ttymgmt',
TTY MANAGEMENT

Enter the name of the new tty line setting [?, q]: 1200300
Select a baud rate [?, q]: ?   (To ask for HELP)

Available baud rates:
    50      134     200       600     1800      4800     19200
    75      150     300      1200     2400      9600     38400
   110
Select a baud rate [?, q]: 1200

Enter the login prompt you want (default = "login: ") [?, q]: Return

Do you want to add another tty line setting to the sequence? [y, n, q] y

Enter the name of the new tty line setting [?, q]: 3001200
Select a baud rate [?, q]: 300
Enter the login prompt you want (default = "login: ") [?, q]: Return
Do you want to add another tty line setting to the sequence? [y, n, q] n

Here is the tty line setting sequence you created:

1200300      3001200
Line Setting:        1200300
    Initial Flags:    B1200 HUPCL
    Final Flags:      B1200 SANE IXANY HUPCL TAB3
    Login Prompt:     login:
    Next Setting:     3001200
Line Setting:         3001200
    Initial Flags:    B300 HUPCL
    Final Flags:      B300 SANE IXANY HUPCL TAB3
    Login Prompt:     login:
    Next Setting:     1200300
B1200    1200 Baud
B 300    300 Baud
HUPCL    Hang Up on Last Close
IXANY    Enable Any Character to Restart Output
SANE     Set All Modes To "Traditionally Reasonable" Values
TAB3     Expand Horizontal-tab To Spaces
Do you want to install this sequence? [y, n, q] y
Installed.
```

P6.3 Modify TTY Line Characteristics

Purpose	To modify TTY line settings or to turn a line on or off
Starting Conditions	System state—multiuser or single-user Login—an authorized login
sysadm menu	TTY MANAGEMENT
Commands	**sysadm modtty**(1)
Reference	Chapter 6, "TTY Management"

In this example, we will tell the computer the port to use with the line settings defined in Procedure P6.2.

Enter this command to go directly to the **modtty** display:

```
$ sysadm modtty
Password:
```

The sequence of prompts shown on the next page then appears on your terminal.

```
Running subcommand 'modtty' from menu 'ttymgmt',
TTY MANAGEMENT

Select the tty you wish to modify,
or enter ALL to see a report of all ttys [?, q]: ttyGA/GAAH

ttyGA/GAAH:  current characteristics:
     State            off
     Hangup Delay     off
     Line Setting     9600
     Description

Available states:
     off     on
Select a state (default: off) [?, q]:  on

Enter a hangup delay in seconds, or 'off' (default:off) [?,q]: 45

(Because this is a dial-up line, we want to specify a timeout figure.)

Available line settings:

(The line settings available on your system appear here.)

Select a line setting (default: 9600) [?, q]:  1200300

Current description:

Enter a new description (default: current description) [?, q]:
1200/300 baud dial in line

ttyGA/GAAH: new characteristics:
     State          on
     Hangup Delay   45
     Line Setting   1200300
    Description     1200/300 baud dial in line
Do you want to install these new characteristics? [y, n, q] y
ttyGA/GAAH now has new characteristics.
```

Chapter P7
Basic Network Procedures

P7.1 Set Up Basic Networking Files P7-1
Set Up ***Devices*** File: **devicemgmt** P7-3
Set Up ***/etc/inittab***: portmgmt P7-5
Set Up ***Systems*** File: systemmgmt P7-7
Set Up ***Poll*** File: pollmgmt P7-9
Set Up ***Permissions*** File P7-11
Set Up ***Devconfig*** File P7-11
Set Up ***Sysfiles*** File P7-12
Add UUCP Logins P7-12
Other Networking Files P7-13

P7.2 Basic Networking Maintenance P7-14
Automated Networking Maintenance (cron) P7-14
uudemon.poll P7-14
uudemon.hour P7-15
uudemon.admin P7-15
uudemon.cleanup P7-16
Manual Maintenance P7-17

P7.3 Basic Networking Debugging P7-18
Check for Faulty ACU/Modem P7-18
Check ***Systems*** File P7-19
Debug Transmissions P7-19
Check Error Messages P7-20
Check Basic Information P7-20

Chapter P7
Basic Network Procedures

This chapter contains the procedures you can use to set up and maintain the Basic Networking system. Chapter 7 in the second half of this manual provides more detailed information about Basic Networking.

This chapter describes the following procedures:

- Configuring basic networking files
- Maintaining basic networking files and operations
- Tracking down problems in basic networking

P7.1 Set Up Basic Networking Files

Purpose	To configure basic networking files and to ensure proper communications links
When Performed	Initial setup and when adding new devices or remote systems
Starting Conditions	System state—2 (multiuser) or 1 (single-user) (You must execute **mount /usr** to run this procedure in single-user mode) Login—an authorized login
sysadm menu	PACKAGE MANAGEMENT
Commands	**sysadm uucpmgmt**(1) **sysadm devicemgmt**(1) **sysadm portmgmt**(1) **sysadm systemmgmt**(1) **sysadm pollmgmt**(1)
Reference	Chapter 7, "Basic Networking"

This procedure provides instructions for setting up the Basic Networking facility and putting it into operation. You will need to use both **sysadm** subcommands and a text editor to carry out this procedure.

The following steps show how to add entries to three of the necessary support files: *Devices*, *Systems*, and *Permissions*. They also show how to modify existing entries in the */etc/inittab* file. This section also describes the setup of several optional files.

To begin, type the command shown on the following screen and the Basic Networking Utilities Management Menu will appear.

```
$ sysadm uucpmgmt
Password:

              BASIC NETWORKING UTILITIES MANAGEMENT

 1 devicemgmt     manage devices (list, add, delete)
 2 pollmgmt       manage poll entries (list, add, delete)
 3 portmgmt       manage I/O ports (list, modify)
 4 systemmgmt     manage remote systems entries (list, add, delete, call)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

You can access the subcommands on this menu either from the menu or directly from the shell.

P7.1.1 Set Up *Devices* File: devicemgmt

The *Devices* file (*/usr/lib/uucp/Devices*) contains information about the devices used to call other machines. For details on this file, refer to Chapter 7.

To add entries to the *Devices* file, type **sysadm devicemgmt**, then select the add option.

```
$ sysadm devicemgmt
Password:

Running subcommand 'devicemgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list, add, and delete entries
in the Basic Networking Utilities '/usr/lib/uucp/Devices' file.
This file contains information about devices
available for calling out using the commands: uucp, cu, and ct.

Type 'q' at any time to quit the present operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question type RETURN for the default.

Enter the operation you want to perform:

1 list
2 add
3 delete

(default list)[q]: 2
```

The following prompts appear:

`port name`
Enter the name of the port to which the device will be connected.

`device name`
Enter the name of the device you are connecting to the port. (A list of available device types appears on the screen; the default value is `penril`.) If you specify **ACU** as the device type, two entries are created, one for 300 baud and one for 1200 baud.

The information you specified then appears on the screen, giving you a chance to change it before it is entered into the *Devices* file.

The */etc/inittab* file might not contain a correct entry for the port just assigned. You can change the port now or later using the **portmgmt** subcommand described in the next procedure.

P7.1.2 Set Up */etc/inittab*: portmgmt

The */etc/inittab* file contains information about the ports to which devices are connected. For further information on this file, refer to Chapter 3, "Processor Operations."

To add entries to the */etc/inittab* file, type **sysadm portmgmt**, then select the `modify` subcommand.

```
$ sysadm portmgmt
Password:

Running subcommand 'portmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list and modify
the entries that control the direction of traffic
on the Basic Networking Utilities I/O ports used by uucp, cu, and ct.

Type 'q' at any time to quit the present operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.

Enter the operation you want to perform:

1 list
2 modify

(default list)[q]:  2
```

The following prompts then appear on your screen.

`port name`
Enter the name of the port you want to modify (the port must be included in the list appearing on your screen).

`traffic direction`
Specify the direction of the traffic on the port. Your choices are **incoming** only, **outgoing** only, or **bidirectional**.

`baud rate`
Enter the speed (baud rate) of the port.

The information you specified then appears on the screen, giving you a chance to change it before it's added to the */etc/inittab* file.

NOTE

*Because adding a device (***sysadm devicemgmt***) automatically creates a port entry in /etc/inittab, you might need to use* **sysadm portmgmt** *only for modifications.*

P7.1.3 Set Up *Systems* File: systemmgmt

The */usr/lib/uucp/Systems* file contains the information **uucp** uses to call and then log onto a remote machine. Each entry represents one remote machine that can be called by your Basic Networking programs.

If the *Systems* entry is to be used to contact a machine that is hardwired to your computer, refer to Chapter 7 for special instructions on setting up the *Systems* file.

To add other machines to your *Systems* file, type **sysadm systemmgmt**, then type **2** to select the `add` subcommand.

```
$ sysadm systemmgmt
Password:

Running subcommand 'systemmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list, add, and delete entries in the
Basic Networking Utilities '/usr/lib/uucp/Systems' file.
This file contains information about what remote systems
can be called by cu and uucp commands.
You can also try to call a remote system that appears
in the '/usr/lib/uucp/Systems' file.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <RETURN> for the default.


Enter the operation you want to perform:

1 list
2 add
3 delete
4 call

(default list)[q]:  2
```

The following prompts then appear on your screen.

`node name`
Type the node name of the system you want to call.

`device type`
Specify the type of device used to establish a connection (for example, **acu**).

`baud rate`
Enter the speed at which the device will place the call.

`phone number`
Enter the telephone number of the remote machine. Special symbols can be embedded in the phone number, including abbreviations from the *Dialcodes* file (*/usr/lib/uucp/Dialcodes*).

`login ID`
Specify the login **uucp** will use to log in on the remote machine.

`password`
Specify the password associated with the above login.

The information you entered will appear on the screen, giving you a chance to change it before it's added to the *Systems* file.

P7.1.4 Set Up *Poll* File: pollmgmt

The */usr/lib/uucp/Poll* file contains a list of the machines that are to be called (polled) by your computer to see if they have anything to transmit to you. It also contains the times they are to be polled. To add entries to the *Poll* file, type **sysadm pollmgmt**, then type **2** to select the `add` option.

```
$ sysadm pollmgmt
Password:

Running subcommand 'pollmgmt' from menu 'uucpmgmt',
BASIC NETWORKING UTILITIES MANAGEMENT

This procedure is used to list, add, and delete entries in the
Basic Networking Utilities '/usr/lib/uucp/Poll' file.
This file contains information about what systems and
the times (hours) the systems should be polled.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type RETURN for the default.


Enter the operation you want to perform:

1 list
2 add
3 delete

(default list)[q]: 2
```

The following prompts then appear on your screen.

`system name`
Type the name of the system you want to poll.

`polling hours`
Specify the hours you want to poll the system; this must be an integer between 0 and 23 (for example, the entry **0 4 8 12 16 20** indicates the system is to be polled every four hours).

The information you entered appears on the screen, giving you a change to change it before it is added to the *Poll* file.

P7.1.5 Set Up *Permissions* File

The default */usr/lib/uucp/Permissions* file provides the maximum amount of security for your computer. The file, as delivered, contains the following entry:

```
LOGNAME=nuucp
```

You can create separate entries for each machine to define these parameters:

- The ways it can receive files from your machine
- The directories in which it can read and write
- The commands it can use for remote execution

See Chapter 7 for information on how to set up this file. If you want to modify the file, use an editor such as **vi**.

P7.1.6 Set Up *Devconfig* File

The */usr/lib/uucp/Devconfig* file is needed only if you are using the Basic Networking Utilities over a Streams-based transport provider that conforms to the AT&T Transport Interface (TI).

You must also create an entry for the transport provider in your *Devices* file. The descriptions in the *Devices* file tell how to define Transport Interface devices.

The entries in the *Devconfig* file define the STREAMS modules used for a particular TI device. (The *push=* variable shows the modules and the order in which they are pushed onto a stream.) Different modules and devices can be defined for **cu** and **uucp** services. If you want to change the contents of this file, use an editor such as **vi**.

P7.1.7 Set Up *Sysfiles* File

The */usr/lib/uucp/Sysfiles* file lets you assign different files to be used by **uucp** and **cu** as *Systems*, *Devices*, and *Dialers* files. Here are some cases where this optional file may be useful:

- You might want different *Systems* files so requests for **cu** login services can be made to addresses other than **uucp** services.
- You might want different *Dialers* files to use different chat scripts for **cu** and **uucp**.
- You might want to have multiple *Systems*, *Dialers*, and *Devices* files. The *Systems* file in particular can become large, making it convenient to split it into several smaller files.

The format of the *Sysfiles* file is described in Chapter 7. Following is an example of the file.

```
service=uucico  systems=Systems.cico:Systems
                dialers=Dialers.cico:Dialers
                devices=Devices.cico:Devices
service=cu      systems=Systems.cu:Systems
                dialers=Dialers.cu:Dialers
                devices=Devices.cu:Devices
```

If you want to modify the contents of this file, use an editor such as **vi**.

P7.1.8 Add UUCP Logins

You must add one or more administrative logins to your system so that incoming **uucp** (**uucico**) requests from remote machines can be handled properly. Each remote machine should have an entry for your machine in its *Systems* file. This entry should contain the login ID and password that you added to your */etc/passwd* file.

Following is a sample entry from the */etc/passwd* file.

```
nuucp:x:6:1:UUCP.Admin:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

This entry shows that a login request by **nuucp** is answered by **/usr/lib/uucp/uucico**. The home directory is */usr/spool/uucppublic*. To assign a password to the login, use **passwd nuucp**.

P7.1.9 Other Networking Files

There are three other files that affect the use of **uucp**. In most cases, the default values are fine and no changes are needed. If you want to change them, however, use an editor such as **ed** or **vi**.

Maxuuxqts
: This file defines the maximum number of **uuxqt** programs that can run at once.

Maxuuscheds
: This file defines the maximum number of **uusched** programs that can run at once.

remote.unknown
: This file is a shell script that executes when an unknown machine starts a conversation. It logs the conversation attempt and fails to make a connection. (If you change the permissions of this file so that it cannot execute, your system will accept any conversation requests.)

P7.2 Basic Networking Maintenance

Purpose	To keep files related to basic networking from consuming too much disk space
When Performed	Automatically with **cron**(1M) or as needed
Starting Conditions	System state—2 (multiuser) or 1 (single-user)

The Basic Networking Utilities come with four shell scripts that poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. These shell scripts should be executed regularly to keep your basic networking running smoothly. Normally, they are run automatically with **cron**(1M), although they can also be run manually. The few areas needing clean up that are not handled by these shell scripts should be maintained manually.

P7.2.1 Automated Networking Maintenance (cron)

UUCP is delivered with entries for **uudemon** shell scripts in the */usr/spool/cron/crontabs/root* file. These entries automatically handle some administrative tasks for you. Each of these shell scripts is in */usr/lib/uucp*.

When the computer is in run state 2 (multiuser), **cron** scans the */usr/spool/cron/crontabs/root* file every minute for entries scheduled to execute at that time. As the UUCP administrator, you should become familiar with **cron** and the four **uudemon** shell scripts described in the following paragraphs.

P7.2.2 uudemon.poll

The **uudemon.poll** shell script, as delivered, performs these tasks:

- Reads the *Poll* file (*/usr/lib/uucp/Poll*) once an hour.
- If any of the machines in the *Poll* file are scheduled to be polled, a work file (*C.sysnxxxx*) is placed in the */usr/spool/uucp/nodename* directory, where *nodename* is replaced by the name of the machine.

This script runs as a daemon process once an hour (by default) and is controlled by this entry in the */usr/spool/cron/crontabs/root* file.

```
40 * * * * /bin/su uucp -c "/usr/lib/uucp/uudemon.poll> /dev/null"
```

P7.2.3 uudemon.hour

The **uudemon.hour** shell script performs these tasks:

- Calls the **uusched** program to search the spool directories for work files (*C.*) that have not been processed and schedules these files for transfer to a remote machine.
- Calls the **uuxqt** daemon to search the spool directories for execute files (*X.*) that have been transferred to your computer and were not processed at the time they were transferred.

The **uudemon.hour** script runs as a daemon process twice an hour (by default) just after the **uudemon.poll** script runs. The following entry in the */usr/spool/cron/crontabs/root* file controls this script.

```
26,56 * * * * /bin/su uucp -c "/usr/lib/uucp/uudemon.hour> /dev/null"
```

P7.2.4 uudemon.admin

The **uudemon.admin** shell script, as delivered, performs these tasks:

- Runs the **uustat** command with the **–p** and **–q** options. The **–q** option reports on the status of queued work files (*C.*), data files (*D.*), and execute files (*X.*). The **–p** option prints process information for networking processes listed in the lock files (*/usr/spool/locks*).
- Sends resulting status information to the **uucp** administrative login via mail.

If you want **cron** to run this script automatically as a daemon process, add an entry such as the following to the */usr/spool/cron/crontabs/root* file.

```
48 11,14 * * 1-5 /bin/su uucp -c "/usr/lib/uucp/uudemon.admin >
/dev/null 2>&1"
```

P7.2.5 uudemon.cleanup

As delivered, the **uudemon.cleanup** shell script performs these tasks:

- Takes log files for individual machines from the */usr/spool/uucp/.Log* directory, merges them, and places them in the */usr/spool/uucp/.Old* directory with other old log information. If log files grow large, you might need to increase the *ulimit*.
- Removes work files (*C.*) and data files (*D.*) that are at least seven days old and execute files (*X.*) that are at least two days old from the spool files.
- Returns undeliverable mail to the sender.
- Mails a summary of the status information gathered during the current day to the UUCP administrative login (**uucp**).

If you want **cron** to run this script automatically as a daemon process, add an entry such as the following to the */usr/spool/cron/crontabs/root* file.

```
45 23 * * * ulimit 5000; /bin/su uucp -c "/usr/lib/uucp/uudemon.cleanup >
/dev/null 2>&1"
```

P7.2.6 Manual Maintenance

Some files can grow indirectly from **uucp** and other basic networking activities. Here are two files you should check and delete if they become too large.

/usr/adm/sulog
: This file keeps a history of all superuser commands. Since the **uudemon** entries in the */usr/cron/root* file use the **su** command, the *sulog* file will grow over time. You should delete this file if it becomes too large.

/usr/lib/cron/log
: This file is a log of **cron** activities. While it grows with use, it is automatically truncated when the system goes to the multiuser state.

P7.3 Basic Networking Debugging

Purpose	To use available monitoring tools to solve basic networking problems
Starting Conditions	System state—2 (multiuser) or 1 (single-user)
Commands	**uustat**(1) **cu**(1) **Uutry**(1) **uuname**(1M) **uulog**(1) **uucheck**(1)

These procedures describe how to solve common UUCP problems.

P7.3.1 Check for Faulty ACU/Modem

The following commands can help you determine if automatic call units or modems are working properly.

- Run **uustat –q**. This command gives counts and reasons for contact failure.
- Run **cu –d –l***line*. This command lets you call over a particular line and prints debugging information on the attempt. For a direct connection, the line must be defined as `Direct` in the *Devices* file. You must add a telephone number to the end of the command line if the line is connected to an autodialer.

P7.3.2 Check *Systems* File

If you are having trouble contacting a particular machine, make sure the information in your *Systems* file is current. The following information might be out of date:

- Phone number
- Login
- Password

P7.3.3 Debug Transmissions

If you are unable to contact a particular machine, you can use **Uutry** and **uucp** to check out communications to that machine.

To simply try to make contact, run the following command, replacing *machine* with the node name of the machine you are having problems contacting.

```
$ /usr/lib/uucp/Uutry -r machine
```

This command performs these tasks:

1. Starts the transfer daemon (**uucico**) with debugging. You will get more debugging information if you are **root**.
2. Directs the debugging output to */tmp/machine*.
3. Prints the debugging output to your terminal (**tail -f**). Press Break to end output.

You can copy the output from */tmp/machine* if you want to save it.

If **Uutry** doesn't isolate the problem, try to queue a job by running the following command, replacing *file* with the file you want to transfer, *machine* with the name of the remote machine, and *dir/file* with the file to be placed on the remote machine.

```
$ uucp -r file machine!/dir/file
```

The **-r** option will queue a job but not start the transfer.

Now use **Uutry** again. If you still cannot solve the problem, call support personnel. Save the debugging output; it will help in diagnosing the problem.

P7.3.4 Check Error Messages

There are two types of error messages for Basic Networking Utilities:

ASSERT Error Messages
: When a process is aborted, ASSERT error messages are recorded in */usr/spool/uucp/.Admin/errors*. These messages include the file name, *sccsid*, line number, and text. These messages usually result from system problems.

STATUS Error Messages
: Status error messages are stored in the */usr/spool/uucp/.Status* directory. The directory contains a separate file for each remote machine with which your computer attempts to communicate. These files contain status information on the attempted communication and tell whether it was successful.

P7.3.5 Check Basic Information

You can use the following commands to check UUCP information.

uuname Lists the systems your machine can contact.

uulog Displays the contents of the log directories for specific hosts.

uucheck –v Checks for the presence of files and directories needed by **uucp**; checks the *Permissions* file and outputs information on the permissions you have set up.

Chapter 1
System Security

1.1 Introduction 1-1

1.2 Security Guidelines 1-1

1.3 Logins and Passwords 1-2
Password Aging 1-2
Specify Password Aging in the */etc/shadow* File 1-3
Specify Password Aging in the */etc/passwd* File 1-3
Locking Unused Logins 1-6
Special Administrative Logins 1-7

Tables

1-1 Password Aging Character Codes 1-4

Chapter 1
System Security

1.1 Introduction

This chapter contains information that can help you maintain the security of your computer system. Chapter P1 in the first half of this guide provides the procedures you'll need to manage your system's identification and security setup.

This chapter includes the following information:

- Security guidelines for setting passwords and permissions and for protecting the system from unauthorized access
- A description of the password aging facility, which allows you to specify when passwords must be changed
- A procedure to lock unused logins
- A description of special administrative commands and logins

1.2 Security Guidelines

The security of the system is the responsibility of everyone having access to it. Although no system is totally secure, the following suggestions can improve your system security:

- Restrict physical access to the machine to authorized users only.
- Set the access permissions for directories and files to allow only the necessary permissions for owner, group, and others.
- Require a password for all logins. Encourage users to change passwords regularly and to pick passwords that aren't obvious. (Nonsense strings that contain 6 to 8 characters and include both letters and numbers are recommended.) Remove or lock unneeded logins.
- Require logins for all dial-up ports.

- Do not keep top secret information on a system with dial-up ports. Any system with dial-up ports is not really secure.
- Check the */usr/adm/sulog* file to monitor usage of the **su** command. To use the **su** command, you must know another user's password. The more people who know a given login and password, the less secure access is to the system. To allow you to monitor **su** command usage, the system keeps a log (*/usr/adm/sulog*) that records its use. Appendix A, "Administrative Directories and Files," describes the format of this log.
- Check the permission levels of login directories, *.profile* files, and files in */bin*, */usr/bin*, and */etc*. These files should not be writable by others.
- Encrypt sensitive data files. The **crypt**(1) command, along with the encryption capabilities of the **ed** and **vi** editors, provides protection for sensitive information. (The **crypt** command is available only in the United States.)
- Log off the system if you must be away from your terminal. Do not leave a logged-in terminal unattended, especially if you are logged in as **root**.

1.3 Logins and Passwords

This section tells how to set up password aging and includes some sample entries in the */etc/passwd* file. It also tells how to lock unused logins and describes some special administrative logins that should be assigned passwords.

1.3.1 Password Aging

The password aging mechanism forces users to change their passwords periodically. You can specify both the minimum and maximum number of days a password can be in effect. Realistically, password aging forces a user to adopt at least two passwords.

If your site is using the */etc/shadow* file to store password information, add the password aging information to that file. If you are not using the */etc/shadow* file, add the password aging information to the */etc/passwd* file.

Specify Password Aging in the */etc/shadow* File

The */etc/shadow* file contains a line for each user set up on your system. Each line contains five fields separated by colons:

username:*password*:*lastchanged*:*min*:*max*

The first field contains the user's login name, the second field contains an encrypted version of the user's password. The *lastchanged* field contains the number of days between January 1, 1970, and the date the password was last changed.

The last two fields, *min* and *max*, contain the password aging information. To add aging information to this file, use an editor such as **vi** to edit the file. Then add the minimum number of days a password can be in effect to the *min* field; add the maximum number of days a password can be in effect to the *max* field.

The system compares these values with the entry in the *lastchanged* field. When a user attempts to change a password, the system verifies that the password has been in effect the number of days specified by the *min* field. If it has not, the system does not allow the user to change the password.

When a user logs in, the system checks the number of days the password has been in effect. If it has been in use for the number of days specified by the *max* field, the system prompts the user to select a new password.

Specify Password Aging in the */etc/passwd* File

If your site is not using the */etc/shadow* file, you must add password aging information to the */etc/passwd* file. This file contains a line for each user set up on the system. Each line contains seven fields separated by colons:

login:*password*:*uid*:*gid*:*account*:*homedirectory*:*program*

The *login* field contains the user's login name, the *password* field contains an encrypted version of the user's password, the *uid* field contains the user's ID number, and the *gid* field contains the number assigned to the user's group. The *account* field contains miscellaneous information about the user, the *homedirectory* field contains the user's home directory, and the *program* field contains the user's shell program.

To add password aging information for a particular user, use an editor such as **vi** to edit the file. Then append the aging information to the password field.

The password aging information consists of a comma and up to four characters in this format:

,Mmww

The characters have the following meanings:

, The delimiter between the password itself and the aging information

M The Maximum duration of the password (in weeks, following the notation in Table 1–1)

m The minimum time interval before the user can change the existing password (in weeks, following the notation in Table 1–1)

ww The week (counted from the beginning of 1970) when the password was last changed (the system automatically adds these characters to the password aging information)

All times are specified in weeks (0 through 63) by a 64 character alphabet. Table 1–1 shows the relationship between the numerical values and character codes. Any of the character codes can be used in the password aging information.

Table 1-1
Password Aging Character Codes

Character	Number of Weeks
. (period)	0 (zero)
/ (slash)	1
0 through 9	2 through 11
A through Z	12 through 37
a through z	38 through 63

Two special cases apply:

- If *M* and *m* are equal to zero (the code ..), the user is forced to change the password at the next login. No further password aging is then applied to that login.

- If *m* is greater than *M* (for example, the code *./*), only **root** can change the password for that login.

Password aging can be set up in a variety of ways to meet the needs of different organizations. Following are some examples.

Change Every Two Weeks

The password aging information in this example requires the user to establish a new password every 2 weeks (0). It also prevents the user from changing the new password for at least 1 week (/).

Here is the original entry in the */etc/password* file for user *jqu*:

```
jqu:RTKESmMOE2m.E:100:1:J.Q. Username:/user1/jqu:
```

To cause *jqu* to change the password at least every two weeks but keep a password at least one week, use the code `0/`. After you edit the */etc/passwd* file, adding `,0/` to the password field, the entry looks like this:

```
jqu:RTKESmMOE2m.E,0/:100:1:J.Q. Username:/user1/jqu:
```

This entry requires *jqu* to change the password at the next login and every two weeks thereafter.

The first time *jqu* logs in after changing the password, the system automatically adds the two-character, "last-time-changed" information to the password field.

```
jqu:RTKESmMOE2m.E,0/W9:100:1:J.Q. Username:/user1/jqu:
```

In this example, *jqu* changed the password in week `W9`.

Change One Time Only

This example requires the user to change the password at the next login.

Here is the orignal */etc/passwd* entry for user *jqu*:

```
jqu:RTKESmMOE2m.E:100:1:J.Q. Username:/user1/jqu:
```

To cause *jqu* to change the password at the next login, you should use the `..` code. (This change is required one time only.) After you edit the */etc/passwd* file, adding `,..` to the password field, the entry looks like this:

```
jqu:RTKESmMOE2m.E,..:100:1:J.Q. Username:/user1/jqu:
```

At the next login, *jqu* will need to change the password.

After *jqu* supplies the new password, the system automatically removes the aging code (, . .) from the password field:

```
jqu:EFDNLqsFUj.fs:100:1:J.Q. Username:/user1/jqu:
```

Note that the encrypted password information has changed.

Change by root Only

The password aging information in this example allows only the **root** login to change the password (the code is . /).

Here is the original */etc/passwd* entry for user *jqu*:

```
jqu:RTKESmMOE2m.E:100:1:J.Q. Username:/user1/jqu:
```

To allow only the **root** login to change *jqu*'s password, use the . / code. After you edit the */etc/passwd* file, adding , . / to the password field, the entry looks like this:

```
jqu:RTKESmMOE2m.E,./:100:1:J.Q. Username:/user1/jqu:
```

Now only **root** can change the password for the *jqu* login. If *jqu* tries to change the password, a "permission denied" message will appear.

1.3.2 Locking Unused Logins

If a login is not used or needed, you should take one of these actions:

- Remove the entry for that login from */etc/passwd*.
- Disable (lock) the login.

If you need to disable a login, use an editor such as **vi** to edit the the encrypted password field. If your site is using the */etc/shadow* file, edit the password field in that file. If you are not using */etc/shadow*, edit the password field in the */etc/passwd* file.

In the password field, type one or more characters that are not used by the encryption process. For example, you could replace the encrypted password with the expression **Locked;**. This expression disables the login because the encryption process does not use the semicolon (;) character. (The word **Locked** serves only to remind you that the login is locked.) You could also type the expression **not valid**, because the encryption process does not use a space.

In the following entry from the */etc/passwd* file, we locked user *jqu's* login.

```
jqu:Locked;:100:1:J.Q. Username:/user1/jqu:
```

1.3.3 Special Administrative Logins

There are two familiar ways to access the system: through a conventional user login or the **root** login. If these were the only ways to access the system, its effective use would be curtailed. Either **root** would own many directories (requiring many users to know the **root** password, which is a bad security risk), or the system would be wide open (because **root** would own few directories). Both of these conditions are undesirable.

A good mix of system use and system security is available through the use of special system logins and administrative commands. The administrative commands (which are also logins) perform functions that might be needed by a number of users on your system:

setup	Sets up the computer. Once the machine has been set up, you do not want anyone doing it again without your knowledge.
sysadm	Allows access to many useful administrative functions that do not require a user to log in as **root**.
checkfsys	Initiates a filesystem check on the specified filesystems.
mountfsys	Mounts the specified filesystem for use.
umountfsys	Unmounts the specified, previously mounted filesystem.
powerdown	Brings the system down to the firmware monitor, where it is safe to turn off the power to the system.
bupsched	Sets up a backup reminder schedule.

These commands allow access to selected directories and system functions. They can be used as login names at the console as well as commands. If you log in to the system with one of these names, the system will execute the command after the login is completed. When you quit or complete the function performed by the command, the system will exit to the `login:` prompt.

Because most of these system functions allow a user access to critical portions of the operating system, you should assign passwords to the administrative commands. Any user attempting to use one of these commands as a login (and any user attempting to execute one of these commands from the shell) will then be prompted for the password.

Following is a list of system logins and the directories they control. You should assign passwords to the **root**, **uucp**, **nuucp**, and **lp** logins. The remaining logins are locked and cannot be used to log into the system.

root This login overrides all other logins, protections, and permissions, giving the user access to the entire operating system. The password for this login should be carefully protected.

sys This login has the power of a normal user login over the files it owns, which are in */usr/src*.

bin This login has the power of a normal user login over the files it owns, which are in */bin*.

adm This login has the power of a normal user login over the object files it owns, which are in */usr/adm*.

uucp This login owns the object and spooled data files in */usr/lib/uucp*.

nuucp This login is used by remote machines to log into the system and initiate file transfers.

daemon This login belongs to the system daemon, which controls background processing.

lp This login owns the object and spooled data files in */usr/spool/lp*.

Chapter 2
User Services

2.1 Introduction 2-1

2.2 Login Administration 2-1
The */etc/passwd* File 2-2
The */etc/shadow* File 2-3
Changing or Deleting Entries in Password Files 2-4
The */etc/group* File 2-5

2.3 The User Environment 2-6
Environment Variables 2-6
Start-Up Files 2-7
System Start-Up Files 2-7
Individual Start-Up Files 2-8
The **umask** Command 2-8
Restricted Shell Program 2-9

2.4 User Communications Services 2-10
Message of the Day 2-10
News Facility 2-10
Broadcast Messages to All Users 2-11
mail and **mailx** 2-12

2.5 Anticipating User Requests 2-12
Trouble Reporting 2-12

Figures

2-1 Environment array for user abc 2-6
2-2 A sample */etc/profile* 2-7

Chapter 2
User Services

2.1 Introduction

This chapter describes the user and group information that must be maintained on the system and discusses user communication services. Chapter P2 in the first half of this manual provides the procedures you can use to maintain user and group information and to broadcast messages to users.

This chapter describes a variety of user services provided by the system administrator:

- Login administration

  Setting up accounts for new users; maintaining the */etc/passwd*, */etc/shadow*, and */etc/group* files

- User environment setup

  Setting up a master start-up file and helping users develop individual start-up files

- User communications services

  Establishing and maintaining such services as message-of-the-day, news, and mail

- User requests

  Developing an organized plan for responding to user problems

2.2 Login Administration

This section describes the */etc/passwd*, */etc/shadow*, and */etc/group* files. The */etc/passwd* file contains information for each user on your system; this information is called a login or user account. The */etc/shadow* file is optional. If your site is using this file, it will store the password information associated with each user account. The */etc/group* file stores information about each group set up on your system.

2.2.1 The */etc/passwd* File

Before users can log into your system, they must be listed in the */etc/passwd* file. The **adduser** option on the User Management Menu leads you through a series of prompts that create entries in the password files. Chapter P2 describes this option.

Each entry in the */etc/passwd* file consists of a single line. Following is a sample entry:

```
abc:x:103:123:Allen B. Cipher:/usr/abc:
```

Each line contains seven fields that are separated by colons:

login name — A valid name for logging onto the system. A login name can be from 3 to 6 characters long; the first character must be alphabetic. Typically, users select their own login names.

password — If your site is not using the */etc/shadow* file, this field contains an encrypted form of the password associated with the login name. The encrypted password contains 13 characters; however the actual password can contain up to 8 characters. At least one character must be numeric. This requirement is to discourage users from choosing ordinary words as passwords. When you add a user to the file, you can use a default password such as *passwd9* and then instruct the user to change it at the first login. This field can also include password aging information, as described in Chapter 1, "System Security."

If your site is using the */etc/shadow* file, the encrypted password and the aging information are stored there and this field contains an `x`.

user id — The user ID number (UID) must be between 0 and 50000. It cannot include a comma. Numbers below 100 are reserved; user ID 0 is reserved for the superuser. When you add a user, the menu system does not allow you to specify a number below 100.

group id — The same conditions apply to the group ID number (GID) as to the UID, except that group 1 is reserved for the "other" group.

account There is no required format for this field. Typically it contains information such as the user's name.

home directory This is the directory where the user is placed when logging in. It is the origination point for the user's directory tree and usually has the same name as the user's login.

program This is the name of the shell program invoked at the time the user logs in. If this field is empty, the default program is **/bin/sh**. This field can be used to invoke a special shell such as **/bin/rsh** (restricted shell).

If you inspect the */etc/passwd* file on your system, you will see several commands listed among the user login names. These are commands such as **bupsched** or **sysadm** that can be assigned passwords.

2.2.2 The */etc/shadow* File

This file contains password information for each user on your system. Because only the user **root** can read */etc/shadow*, it provides more security than */etc/passwd*, which all users can read. The system consults */etc/shadow* whenever it needs to verify password information.

Each entry in the */etc/shadow* file contains the following fields:

username The user's login name.

password An encrypted version of the user's password. If the login is disabled, this field also includes a lock string.

lastchanged The number of days between January 1, 1970, and the date the password was last modified.

min The minimum number of days that a password must be in effect.

max The maximum number of days that a password can be in effect.

2.2.3 Changing or Deleting Entries in Password Files

The User Management Menu contains options for adding, changing, or deleting user entries from the password files. (Chapter P2 describes these options.) If you want to update the files manually, use the **passmgmt** command. (See **passmgmt**(1M) in the *Reference Manual.*)

These two methods produce different results when used to remove a user account. If you use the **deluser** command from the User Management Menu to delete a user login, all of the user's files and directories are also removed. If you use the **passmgmt** command to remove a user login, the user's files will not be deleted.

If you need to change a user's password, you must use the **passwd** command. If a user forgets a password or you need to change a password for another reason, log in as **root**. Then issue the **passwd** command, followed by the user's login name. (The # prompt shows you are **root**.)

```
# passwd abc
New password:
Re-enter new password:
```

The new password does not appear on the screen as you type it. Because you ran this command as superuser (**root**), you were not prompted for the old password. After you change a password, ask the user to select a new password immediately.

2.2.4 The */etc/group* File

Groups provide a means to establish ownership and access to files and directories. Users with a community of interest can be identified as members of the same group. Each file created by a member of the group carries the group ID as a secondary identification. By manipulating the permissions assigned to the file, the owner (or someone with the effective user ID of the owner) can grant read, write, or execute privileges to other group members.

The */etc/group* file stores information about the groups set up on your system. There is a separate line for each group. Following is a sample entry from this file:

```
prog::123:jqp,abc
```

Each line contains four fields separated by colons:

group name The group name can be from 3 to 8 characters long; the first character must be alphabetic.

password The password field should not be used.

group id The group ID is a number from 0 to 50000. It must not include a comma. Numbers below 100 are reserved.

login names The login names listed here can use the **newgrp** command to switch to this group.

2.3 The User Environment

This section describes environment variables, tells how the system-wide start-up files and individual start-up files are used, and discusses restricted shell programs.

2.3.1 Environment Variables

An array of strings called the environment is made available by **exec**(2) when a process begins. Because **login** is a process, the array of environment strings is made available to it. Figure 2-1 defines 12 environment variables for user *abc*.

```
PS1=$
LOGNAME=abc
PWD=/usr/abc
HOME=/usr/abc
PATH=:/bin:/usr/bin:/usr/lbin
SHELL=/bin/sh
MAIL=/usr/mail/abc
TERM=5420
PAGER=pg
TZ=EST5EDT
TERMINFO=/usr/lib/terminfo
EDITOR=vi
```

Figure 2-1. Environment array for user abc.

Other programs also use these environment variables. For example, the user's terminal is defined as a 5420 (`TERM=5420`). When the user invokes the **vi** editor, **vi** checks the file (*/usr/lib/terminfo*) referenced by the **TERMINFO** variable, where it learns the characteristics of a 5420 terminal (such as the 24-line screen).

New strings can be defined at any time. By convention, they are defined with the variable in uppercase, followed by an equal sign, followed by the value. After defining an environment variable, the user can use the **export** statement to make the variable global. This means the variable will apply to any subshells started up during the login session.

2.3.2 Start-Up Files

Start-up files are the key element in establishing an environment that allows users to communicate successfully with the computer. There are two types of start-up files: system-wide files that affect all users on the system and individual start-up files that can be different for each user.

System Start-Up Files

There are two system start-up files: the */etc/profile* file contains values that apply to Bourne and Korn shell users; the */etc/csh.login* file applies to C shell users. These files contain commands, shell procedures, and environment variables. Whenever a user logs in, the **login** process executes the files. Figure 2-2 shows a sample */etc/.profile* file.

```
#  The profile that all logins get before using their own .profile.
trap "" 2 3
export LOGNAME
#  Login and -su shells get /etc/profile services.
#  -rsh is given its environment in its .profile.
case "$0" in
-su)
    export PATH
    ;;
-sh)
    export PATH
    #  Allow the user to break the Message of the Day only.
    trap "trap '' 2" 2
    cat -s /etc/motd
    trap "" 2
    if mail -e
    then
        echo "you have mail"
    fi
    if [ ${LOGNAME} != root ]
    then
        news -n
    fi
    ;;
esac
umask 022
trap 2 3
```

Figure 2-2. A sample */etc/profile*.

The sample */etc/profile* contains several interesting entries:

- Some environment variables are exported.
- The **cat** command is executed on a file named */etc/motd*, causing the contents of that file to appear on the user's terminal. (See "User Communications," later in this chapter.)
- If the user is not **root**, the command **news –n** causes the names of news items to appear on the user's terminal. (See "User Communications," later in this chapter.)
- If the user has mail, the command **mail –e** causes a message about the waiting mail to appear on the user's terminal.

For information about the shell programming commands used in Figure 2-2, see **sh**(1) in the *Reference Manual.*

Individual Start-Up Files

Each user can have an individual start-up file that contains additional commands and variables to further customize the user's environment. For Bourne and Korn shell users, the file is called *.profile*. For C shell users, there are two start-files: *.login* and *.cshrc*. These files are located in the user's home directory and can contain whatever the user wants. If a local start-up file exists, it is executed at login time, after the system start-up file is executed.

2.3.3 The umask Command

A system default controls the permissions mode of all files or directories created by a user. The operating system assigns the default values of 666 for files and 777 for directories. That means that everyone automatically has read and write permission on files. For directories, everyone has read, write, and execute permission. (Execute permission on a directory means the ability to move to the directory and to copy files from it.)

Users can include the **umask** command in their individual start-up files. You can also include a **umask** command in the system-wide start-up files. This command alters the default permission levels by the amount specified. For example, the following line leaves the permission level for the owner unchanged, lowers the permission level for the group by 2, and reduces the permissions for others to zero.

```
umask 027
```

The system default was 666; this user mask changes it to 640, which translates into read and write permission for the owner, read permission for the group, and no permission for others.

2.3.4 Restricted Shell Program

When a user logs in, the shell program specified in the */etc/passwd* file starts up. For most users, this will be a standard shell program such as **/bin/sh**. If you want to limit the activities a user can perform, you can assign a restricted shell to the user. This type of shell prevents the user from performing these tasks:

- Changing directories
- Changing the value of the **PATH** environment variable
- Specifying pathnames or command names containing a slash (/). (This means the user can access only the files or directories included in the present working directory or in **$PATH**.)
- Redirecting output

The restrictions are enforced after the user's start-up file executes.

The administrator can use a restricted shell strategy to limit certain users to the execution of a small number of commands or programs. By setting up a special directory for executables (*/usr/rbin*, for example) and setting the **PATH** environment variable so it references only that directory, the administrator can restrict the user's activities as appropriate.

2.4 User Communications Services

The operating system offers several ways to communicate with users. Some of the most frequently used methods are described in this section.

2.4.1 Message of the Day

Information that you want to make available to all users can be placed in the */etc/motd* file. The contents of this file appear on the user's terminal when the user logs in. Typically, the system-wide start-up file contains this command:

cat /etc/motd

When the **login** process executes the start-up file, it prints the text in */etc/motd* on the user's screen.

Following is a typical message:

```
5/30:  The system will be unavailable from 6-11pm
```

For this information to have any impact on users, you must use it sparingly and clean out outdated announcements.

2.4.2 News Facility

The */usr/news* directory and the **news**(1) command also provide a means to communicate with users. The directory is used to store news items in text files. The **news** command is used to print the items on your terminal.

If the system start-up file contains the following line, the names of files in the */usr/news* directory print on a user's terminal when the user logs in.

news -n

Item names appear only for current items, that is, items added to the */usr/news* directory since the user last looked at the news.

When you read a news item, an empty file named *.news_time* is written in your login directory. As with any other file, *.news_time* carries a time stamp indicating the date and time the file was created. When you log in, a comparison is made between the time stamp of your *.news_time* file and time stamp of items in */usr/news*.

The **news** command is used to read news items. Users have a choice of several possible actions:

Read everything
: If a user issues the **news** command with no arguments, all news items posted since the last time the user issued the command print on the user's terminal.

Select some items
: If the **news** command is entered with the names of one or more items as arguments, only the specified items appear.

Read and delete
: After issuing the **news** command, the user can stop any item from printing by pressing Delete. Pressing Delete twice in a row stops the program.

Ignore everything
: If a user doesn't want to read the news items at the present time, the items can safely be ignored. Unless the user deletes the items, the item names will continue to appear each time the user logs in.

Flush all items
: If a user doesn't want to read the news items at all, there are a couple of techniques that will delete the items. The following command updates the time-accessed and time-modified fields of the **.news_time** file.

```
$ touch .news_time
```

The next command prints the news items on the null device.

```
$ news > /dev/null
```

2.4.3 Broadcast Messages to All Users

The **wall** command allows you to write messages to all users who are logged in on the system. When you need to use this command, first become superuser. This is necessary to override the **mesg n** command that many users include in their individual start-up files. (The **mesg n** command prevents messages sent by other users, via the **write** command, from appearing on the user's terminal.) It is best to reserve the **wall** command for those times when you need to ask users to log off the system. Chapter P2 shows an example of the **wall** command.

2.4.4 mail and mailx

The operating system offers two electronic mail utilities that allow users to communicate among themselves. If your system is connected to other systems by networking facilities, **mail** and **mailx** can also be used to communicate with people on other systems.

mail is the basic utility for sending messages. **mailx** uses **mail** to send and receive messages, but adds many other features that are useful for reading, storing, and sending messages.

When **mailx** is used, a set-up file is helpful. You can find a description of a *.mailrc* set-up file under **mailx**(1) in the *Reference Manual.*

2.5 Anticipating User Requests

As system administrator, you can expect users to look to you to help solve a variety of problems. In addition to the system log described in Chapter 3, you will find it helpful to keep a user trouble log. The problems that users run into fall into patterns. If you keep a record of how problems were resolved, you will not have to start from scratch when a problem recurs.

2.5.1 Trouble Reporting

It is strongly recommended that you develop an organized way for users to report problems. The following page contains a sample Trouble Report that can be used to report system problems.

TROUBLE REPORT

Machine _____

Program running _____

Production or development _____

Type _____

Symptoms _____

Scope _____

Error Messages _____

_____

_____

Person reporting _____ login ____

Location _____ Phone _____

Chapter 3
System Operations and Run Levels

3.1 Introduction 3-1
General Operating Policy 3-1
Maintaining a System Log 3-2
Administrative Directories and Files 3-3
Root Directories 3-3
Important System Files 3-3

3.2 Operating Levels 3-6
How **init** Controls the System State 3-8
Entering the Multiuser State 3-10
Powering Up 3-10
Early Initialization 3-10
Preparing the Run-Level Change 3-11
The System Life Cycle 3-12
Changing Run Levels 3-12
Run Level Directories 3-12
Going to Single-User Mode 3-13
Going to Run Levels 5 or 6 3-14
Turning the System Off 3-14

Tables

3-1 System States 3-7

Chapter 3
System Operations and Run Levels

3.1 Introduction

This chapter provides detailed information about day-to-day operations and system run levels. Chapter P3 in the first half of this manual contains the procedures you can use to remove power from the system and to change the system run level.

This chapter includes these topics:

- General operating policy

  Guidelines for balancing the needs of system maintenance and the interests of your user community, suggestions for record keeping, and lists of important administrative directories and files

- Operating Levels

  Definitions of the operating levels and a description of how they are controlled

3.1.1 General Operating Policy

Many administrative tasks require the system to be shut down to a run level other than the multiuser state. When the system is in another run level, regular users cannot access the system. Try to schedule tasks requiring that you change the system state for times when they will interfere the least with the activities of the user community.

Try to provide the user community as much notice as possible about events affecting the use of the machine. When the system must be taken out of service, also tell the users when to expect the system to be available. Use the news facility (**/etc/news**) and the Message of the Day (*/etc/motd*) to keep users informed about changes in hardware, software, policies, and procedures.

At your discretion, the following items should be done as prerequisites for any task that requires the system to leave the multiuser state.

1. When possible, schedule these tasks for periods of low system use. Use the Message of the Day (*/etc/motd*) to inform users of the schedule.
2. Check to see who is logged in before taking any actions that would affect a logged-in user. The **/etc/whodo** and **/bin/who** commands can be used to see who is on the system.
3. If the system is in use, give your users advanced warning about changes in system states or pending maintenance actions. For immediate actions, use the **/etc/wall** command to send a broadcast message announcing that the system will be taken down at a given time. Give the users a reasonable amount of time to terminate their activities and to log off before you take the system down.

3.1.2 Maintaining a System Log

In a multiuser environment, it is strongly recommended that a complete set of records be maintained. A system log book can be a valuable tool when troubleshooting transient problems or when trying to establish system operating characteristics over a period of time. Following are some items you might want to enter into your log book:

- Maintenance records (dates and actions)
- Printouts of error messages and diagnostic phases
- Records of equipment and system configuration changes (dates and actions)

The format of the system log and the types of items it contains should follow a logical structure. Think of the log as a diary that you update on a periodic basis. To a large measure, how you use your system will dictate the form and importance of maintaining a system log.

3.1.3 Administrative Directories and Files

This section describes the directories and files that are frequently used by a system administrator. For more detail about the purpose and contents of these directories and files, see Appendix A. For additional information on the formats of the system files, refer to Section 4 in the *Reference Manual.*

3.1.4 Root Directories

The following directories, which are located in the **root** filesystem (/), are important for system administration.

bin	Contains public commands.
dev	Contains special files that define all of the devices on the system.
etc	Contains administrative programs and tables.
lib	Contains public libraries.
lost+found	Used by **fsck**(1M) to save disconnected files.
mnt	Used to temporarily mount filesystems.
save	Used by the menu system for saving data.
tmp	Used for temporary files.
usr	Used to mount the */usr* filesystem (see Chapter 4 for a description of this filesystem).

3.1.5 Important System Files

The following files and directories are important in the administration of the computer:

/etc/checklist	Defines a default list of filesystem devices to be checked by **/etc/fsck**.
/etc/fstab	Specifies the filesystem(s) to be mounted by **/etc/mountall**.

/etc/gettydefs	Contains information used by **/etc/getty** to set the speed and terminal settings for a line.
/etc/group	Describes each group to the system.
/etc/init.d	Contains executable files used in upward and downward transitions to all system run levels. These files are linked to files beginning with *S* (start) or *K* (stop) in */etc/rcn.d*, where *n* is replaced by the appropriate run level.
/etc/inittab	Contains the instructions to define the processes created or terminated by **/etc/init** for each initialization state.
/etc/motd	File containing a brief Message of the Day, which is output by the system start-up files.
/etc/passwd	File identifying each user to the system.
/etc/profile	File containing the standard (default) environment for all Bourne and Korn shell users.
/etc/rc0	File that executes shell scripts in */etc/rc0.d* and */etc/shutdown.d* directories for transitions to system run-levels 0, 5, and 6. This file is executed by **/etc/shutdown**.
/etc/rc0.d	Directory containing files executed by **/etc/rc0** for transitions to system run-levels 0, 5, and 6. Files in this directory are linked from files in the */etc/init.d* directory and begin with either a *K* or an *S*. *K* indicates processes that are stopped when entering run levels 0, 5, or 6; *S* indicates processes that are started.
/etc/rc2	File, executed by **/etc/init**, that executes shell scripts in */etc/rc2.d* and */etc/rc.d* on transitions to run level 2.
/etc/rc2.d	Directory containing files executed by **/etc/rc2** for transitions to run levels 2 and 3. Files in this directory are linked from files in the */etc/init.d* directory and begin with either a *K* or an *S*. *K* indicates processes that should be stopped when entering run levels 2 or 3; *S* indicates processes that should be started.
/etc/rc.d	Directory containing executable files that carry out the various functions needed to initialize the system to run level 2; they are executed when **/etc/rc2** is run. This directory is maintained only for compatibility.

/etc/save.d Directory containing files that are used by the system administration menu commands associated with backing-up data.

/etc/shutdown File containing a shell script that gracefully shuts down the system in preparation for system backup or for scheduled downtime.

/etc/shutdown.d
Directory containing executable files that carry out the various functions needed to transition the system to the single-user state (run-levels 1, s, or S). This directory is maintained only for compatibility.

/etc/TIMEZONE
File used to set the time zone environment variable **TZ**.

/etc/utmp File containing the information on the current run-state of the system.

/etc/wtmp File containing a history of system logins.

/usr/adm/sulog
File containing a history of **su** command usage. This file should be checked periodically for size.

/usr/lib/cron/log
File containing a history of all actions taken by **/etc/cron**. This file should be checked periodically for size.

/usr/lib/help/HELPLOG
File containing a history of all actions taken by **/usr/bin/help**.

/usr/lib/spell/spellhist
File containing a history of all words that **spell** fails to match.

/usr/news Directory containing news files. This directory should be checked periodically and old files should be discarded.

/usr/options Directory containing files that identify the utilities that are installed on the system.

/usr/spool/cron/crontabs
Directory containing crontab files for the **adm**, **root**, and **sys** logins and for ordinary users listed in *cron.allow*.

Appendix A describes each of these files in more detail.

3.2 Operating Levels

After you have set up your computer for the first time (plugging it in, hooking all the hardware together, installing the system software, booting it, running the setup programs), you and other users can use the system. Whenever you turn it on (including the first time), the system comes up in a multiuser environment. Following are some characteristics of this environment:

- The filesystems are mounted.
- The **cron** daemon is started for scheduled tasks.
- The basic networking functions of **uucp** (if added to the system) are available.
- The LP spooling and scheduling functions are available for use.
- Users can log in. **gettys** are spawned on all connected terminal lines having a **respawn** entry in */etc/inittab*. (**gettys** are not on when the system is installed. You have to turn them on yourself.)

This is defined as the multiuser state. It is also referred to as **init** state 2 because all activities involved in initializing the system are under the control of the **init** process. The "2" refers to entries in the special table */etc/inittab* used by **init** to initialize the system to the multiuser state.

Not all activities, however, can be performed in the multiuser state. For example, if you were able to unmount a filesystem while users were accessing it, a lot of data could be lost. For unmounting filesystems and other system administration tasks, there is a need for another state, the single-user state.

The single-user state is an environment in which only the console has access to the system and only the root filesystem is mounted. You are free to do tasks that affect filesystems and system configuration because you are the only one on the system.

There are other system states (see Figure 3-1), but first a note of clarification. One of the more confusing things about the discussion of system states is that there are many terms used to identify the same thing: the particular operating level of the system.

For example, the system state is also called the run state, the run level, the run mode, or the init state. Similarly, each system state can be referred to in a number of ways. For example, the single-user state is also called single-user mode or run level 1.

In any case, each state or run level clearly defines the operation of the computer. Table 3-1 defines each run level.

Table 3-1
System States

Run Level	Description
0	Power down; takes the system to firmware mode, where it is safe to shut off power to the system. You can also run stand-alone programs.
1, s, or S	Single-user mode is used to install or remove software utilities, run filesystem backups and restores, and check filesystems. Although **s** and **1** both go to single user state, **s** kills only those processes spawned by **init** and does not unmount filesystems. State **1** unmounts everything except root and kills all user processes, except those that relate to the console.
2	Multiuser mode is the normal operating mode for the system. The default is that the *root* (/) and user (*/usr*) filesystems are mounted.
3	User-defined run state.
4	User defined run state.
5	Firmware mode; same as run level 0.
6	This mode is used to halt and reboot the operating system. The system comes up in multiuser mode (run level 2).

3.2.1 How init Controls the System State

The operating system always runs in one state or another. The actions that cause the various states to exist are under the control of the **init** process, which is the first general process created by the system at boot time. It reads the file */etc/inittab*, which defines the processes that exist at each run level.

In the case of the multiuser state (run level 2), **init** scans the file for entries having a tag for the run level (the tag is a 2) and executes everything after the last colon (:) on the line containing the tag. These tags represent the run levels in Table 3-1.

Following are entries from a sample */etc/inittab*. (It is unlikely that your file will look exactly like this example; */etc/inittab* changes from one configuration to another.)

```
bck::sysinit:/etc/bcheckrc >/dev/console 2>&1
bdf::sysinit:/etc/bdflush >/dev/console 2>&1 &
dvb::sysinit:/etc/devbuild -a >/dev/console 2>&1
is:2:initdefault:
so:056:wait:/etc/rc0 >/dev/console 2>&1
s1:1:wait:/etc/rc1 >/dev/console 2>&1
s2:23:wait:/etc/rc2 >/dev/console 2>&1
s3:3:wait:/etc/rc3 >/dev/console 2>&1
fw:5:wait:/etc/uadmin 2 0 >/dev/console 2>&1
RB:6:wait:echo "\nThe system is being restarted." >/dev/console 2>&1
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1
pf:s1234:powerwait:/etc/powerfailpowerfail 1> /dev/console 2>&1
po:s1234:powerokwait:/etc/powerfailpowerok 1> /dev/console 2>&1
pw:234:respawn:/etc/pwdhd >/dev/null </dev/null 2>&1
co:234:respawn:/etc/getty syscon console sqnt200
ac:234:off:/etc/getty altcon console hp700
GAAA:234:respawn:/etc/getty ttyGA/GAAA 9600 sqnt200 #
GAAB:234:respawn:/etc/getty ttyGA/GAAB 9600 sqnt200 #
GAAC:234:respawn:/etc/getty ttyGA/GAAC 9600 sqnt200 #
```

The lines in */etc/inittab* have this format:

id:level:action:process

- *id* is a label, from one to four characters long, that identifies an entry. Four-character *id's* are typically used for entries defining terminal lines.

- *level* is zero or more numbers and letters (**0** through **6**, **s**, **a**, **b**, and **c**) that determine the *level*(s) in which *action* is to take place. If *level* is null, the *action* is valid in all levels.
- *action* can be one of the following:

sysinit	Run *process* before **init** sends anything to the system console (`Console Login:`).
bootwait	Start *process* the first time **init** goes from single-user to multiuser state after the system is booted. (If **initdefault** is set to **2**, the process will run right after the boot.) *init* starts the process, waits for its termination, and, when it dies, does not restart the process.
wait	When going to *level*, start *process* and wait until it's finished.
initdefault	When **init** starts, it will enter *level*; the *process* field for this *action* has no meaning.
once	Run *process* once and don't start it again if it finishes.
powerfail	Tells **init** to run *process* whenever a direct powerdown of the computer is requested.
respawn	If *process* does not exist, start it, wait for it to finish, and then start another.
ondemand	Synonymous with **respawn**, but used only with *levels* **a**, **b**, or **c**.
off	When in *level*, kill process or ignore it.

- *process* is any executable program, including shell procedures.
- A pound sign (#) can be used to add a comment to the end of a line. Everything after a # to the end of the line will be ignored by **init**.

When changing levels, **init** kills all processes not specified for that level.

NOTE

If ***/etc/inittab*** *was removed by mistake and is missing during shutdown,* **init** *will enter the single-user state (***init s***). The* ***/usr*** *filesystem will remain mounted and processes not spawned by* **init** *will continue to run. You should replace* ***/etc/inittab*** *before changing states again.*

3.2.2 Entering the Multiuser State

Powering Up

When you power up your system, it will enter multiuser state by default. (You can change the default by modifying the `initdefault` line in your *inittab* file.) In effect, going to the multiuser state follows these broad lines:

1. You turn on the computer.
2. The operating system is loaded and the early system initializations are started by **init**.
3. The run level change is prepared by the **/etc/rc2** procedure.
4. The system is made public via the spawning of **getty**s along the terminal lines.

Early Initialization

Just after the operating system is first loaded into core via the specialized boot programs (see **boot**(8) in the *Reference Manual* for more detail), the **init** process is created. It immediately scans */etc/inittab* for entries of the type **sysinit**:

```
bck::sysinit:/etc/bcheckrc >/dev/console 2>&1
bdf::sysinit:/etc/bdflush >/dev/console 2>&1 &
dvb::sysinit:/etc/devbuild -a >/dev/console 2>&1
```

They are executed in sequence and perform the necessary early initializations of the system. Note that each entry indicates a standard input/output relationship with */dev/console*. This is the way communication is established with the system console before the system has been brought to the multiuser state.

Preparing the Run-Level Change

Now the system must be placed in a particular run level. First, **init** scans the table to find an entry that specifies an *action* of the type *initdefault*. If it finds one, it uses the run level of that entry as the tag for selecting the next entries to be executed. In our sample */etc/inittab*, the **initdefault** entry specifies run level 2 (the multiuser state) as the tag to select and execute other entries:

```
is:2:initdefault:
s2:23:wait:/etc/rc2 >/dev/console 2>&1
pf:s1234:powerwait:/etc/powerfailpowerfail 1> /dev/console 2>&1
po:s1234:powerokwait:/etc/powerfailpowerok 1> /dev/console 2>&1
pw:234:respawn:/etc/pwdhd >/dev/null </dev/null 2>&1
co:234:respawn:/etc/getty syscon console hp700
ac:234:off:/etc/getty altcon console hp700
GAAA:234:respawn:/etc/getty ttyGA/GAAA 9600 sqnt200 #
GAAB:234:respawn:/etc/getty ttyGA/GAAB 9600 sqnt200 #
GAAC:234:respawn:/etc/getty ttyGA/GAAC 9600 sqnt200 #
```

The other entries shown above specify the actions necessary to prepare the system to change to the multiuser run level. First, **/etc/rc2** is executed. It executes all files in */etc/rc2.d* that begin with the letter *S*. It then executes all the files in the */etc/rc.d* directory, accomplishing (among other things) the following:

- Sets up and mounts the filesystems.
- Starts the **cron** daemon.
- Displays the current system hardware configuration.
- Makes **uucp** available for use, if installed.
- Makes the LP system available for use, if installed.
- Starts a **getty** for the console.
- Starts the hard disk error logging daemon.
- Starts a **getty** on terminal lines.

The full multiuser environment is now established and your system is available for users to log in.

3.2.3 The System Life Cycle

Changing Run Levels

Changing run levels follows these broad lines:

1. Enter a command that directs **init** to execute entries in */etc/inittab* for a new run level.
2. Key procedures such as **/etc/shutdown**, **/etc/rc0**, **/etc/rc2**, and **/etc/rc3** are run to initialize the new state.
3. The new state is reached. If it is either state 1 or 5, you can proceed.

Run Level Directories

Run levels 0, 2, and 3 each have a directory of files that are executed in transitions to and from that level. These directories are *rc0.d*, *rc2.d*, and *rc3.d*, respectively. All files in these directories are linked to files in */etc/init.d*. The run-level filenames look like this:

S00name

or

K00name

The filenames can be split into three parts:

S or *K* — The first letter defines whether the process should be started (*S*) or stopped (*K*) upon entering the new run level.

00 — The next two characters are a number from 00 to 99. They indicate the order in which the files will be started (S00, S01, S02, and so on) or stopped (K00, K01, K02, and so on).

name — The file in */etc/init.d* that this file is linked to.

For example, the *init.d* file *cron* is linked to the *rc2.d* file *S75cron* and *rc0.d* file *K70cron*. When you enter **init 2**, this file is executed with the **start** option: *sh S75cron start*. When you enter **init 0**, this file is executed with the **stop** option: *sh K70cron stop*. This particular shell script will execute **/usr/bin/cron** when run with the **start** option and kill the **cron** process when run with the **stop** option.

Because these files are shell scripts, you can read them to see what they do. You can modify the files, though it is preferable to add your own since the delivered scripts may change in future releases. To create your own scripts you should follow these rules:

- Place the file in */etc/init.d*.
- Link the file to files in appropriate run state directories by using the naming convention described earlier.
- Have the file accept the **start** and/or **stop** options.

Going to Single-User Mode

At times you will need to perform some administrative functions in the single-user mode. The normal way to go to single-user mode is through the **/etc/shutdown** command. This procedure executes all of the files in the */etc/rc0.d* and */etc/shutdown.d* directories by calling the **/etc/rc0** procedure. The following steps are performed during the shutdown:

- Close all open files and stop all user processes.
- Stop all daemons and services.
- Write all system buffers out to disk.
- Unmount all filesystems except root.

The entries for single-user processing in the sample */etc/inittab* are as follows:

```
s1:1:wait:/etc/rc1 >/dev/console 2>&1
pf:s1234:powerwait:/etc/powerfailpowerfail 1> /dev/console 2>&1
po:s1234:powerokwait:/etc/powerfailpowerok 1> /dev/console 2>&1
```

There are two major ways to start the shutdown processing:

- Enter the **shutdown -i1** command (recommended).
- Enter the **init 1** command, which forces the **init** process to scan the table. The first entry it finds is the **s1** entry, which starts the shutdown processing.

Now the system is in the single-user environment and you can perform the appropriate administrative tasks.

Going to Run Levels 5 or 6

To go to the firmware mode or to reboot the system, use the **shutdown** command with the **–i5** or the **–i6** option. Note that the pertinent entries in */etc/inittab* are similar:

```
so:056:wait:/etc/rc0 >/dev/console 2>&1
fw:5:wait:/etc/uadmin 2 0 >/dev/console 2>&1
RB:6:wait:echo "\nThe system is being restarted." >/dev/console 2>&1
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1
```

For both system states, the **/etc/rc0** procedure is called. Then the **/etc/uadmin** procedure, with different arguments for each state, is called. This procedure invokes the **uadmin**(2) system call, which directly prepares either system state.

The computer stays in state 5 as long as you want it to. It stays in state 6 briefly to restart the entire system initialization. Both states are similar, however, since they exist when the computer is under firmware control.

Turning the System Off

The final step in the life cycle of the system is turning it off. The following entry applies to powering the system down.

```
so:056:wait:/etc/rc0 >/dev/console 2>&1
```

One way to turn the system off is to invoke the **/etc/shutdown –i0** command. The **/etc/shutdown** and **/etc/rc0** procedures are called to clean up and stop all user processes, daemons, and other services and to unmount the filesystems. When this is done, you can turn off the power to the system.

Chapter 4
Filesystem Administration

4.1 Introduction 4-1

4.2 Filesystems 4-1
Filesystems and Disks 4-2
Cylinders 4-4
The Filesystem Structure 4-7
Inodes 4-8
Data Blocks 4-8
Fragments 4-9
Indirect Blocks 4-9

4.3 Maintaining Filesystems 4-11
The Need for Policies 4-11
Shell Scripts for Filesystem Administration 4-11
Monitoring Disk Usage 4-12
Monitoring Percent of Disk Space Used 4-12
Monitoring Files and Directories that Grow 4-13
Identifying and Removing Inactive Files 4-13
Identifying Large-Space Users 4-14

4.4 Filesystem Backup and Restore 4-15
Complete Backup 4-15
Incremental Backup 4-16
Backup Schedule Reminder 4-17
Restoring a Filesystem from Backup 4-18

4.5 Checking Filesystems 4-19
What Can Go Wrong with a Filesystem 4-19
Checking a Filesystem for Consistency 4-20
The **fsck** Command 4-20
Fix Corrupted Filesystems 4-21
Detect and Correct Corruption 4-21

Figures

4-1 The *root* filesystem 4-2
4-2 Disk partitions containing the *root* and *usr* filesystems 4-4
4-3 The relationship between disk sectors, tracks, and file blocks 4-5
4-4 Cylinders on a sample disk pack 4-6
4-5 Disk layout 4-7
4-6 Locating disk storage blocks 4-10
4-7 Sample */etc/save.d/except* file 4-17

Chapter 4
Filesystem Administration

4.1 Introduction

This chapter provides information about filesystems and their maintenance. Chapter P4 in the first half of this manual contains procedures you can use to maintain filesystems and to monitor disk use.

This chapter describes filesystems, tells how to monitor disk space use, how to back up and restore filesystems, and how to check a filesystem for directory consistency.

4.2 Filesystems

A filesystem is a combination of directories and files arranged in a hierarchical "tree" structure. A directory is a file that contains other files or directories. Each filesystem begins with a directory that serves as the root for the filesystem. The *root* filesystem is the starting point for the operating system itself. A slash (/) represents the root directory for this filesystem.

You can attach, or *mount*, other filesystems at any point on the already existing tree structure. A filesystem is attached at a directory called a *mountpoint*. Figure 4-1 shows the */usr* directory, which can be used as the mountpoint for the */usr* filesystem. It provides a connection between the root filesystem and the contents of the */usr* filesystem. The filesystem has the same name as its mountpoint (*/usr* in this example).

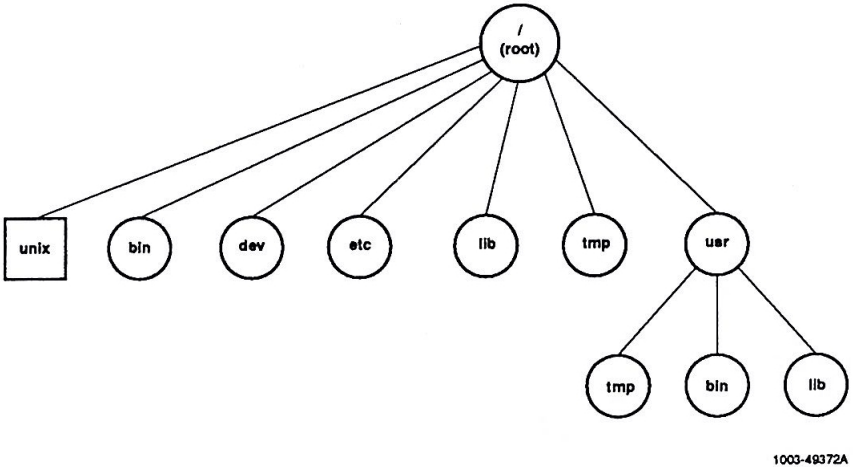


Figure 4-1. The *root* filesystem. *The usr* directory is a mountpoint for the *usr* filesystem, which contains the *tmp*, *bin*, and *lib* directories.

A filesystem must be mounted to allow users access to the files it contains. Users cannot access the files in an unmounted filesystem.

In most cases, the boundaries between filesystems are invisible to users. If users have the appropriate permissions for the files in another filesystem, they can access those files and can execute most operating system commands.

Each directory contains two entries: a dot (.) that represents the current directory, and a dot dot (..) that represents the parent directory. The root directory for each filesystem also contains these entries. When the filesystem is mounted, .. points to the parent of the mountpoint directory. For root itself (the topmost directory), the .. points back to itself.

4.2.1 Filesystems and Disks

Filesystems give you a means to spread files among the disk drives in your system. Each disk can contain one or more filesystems.

To accommodate filesystems, each disk is divided into *partitions* labeled *0*, *1*, *2*, *3*, and so on. The system administrator can specify the number and size of the partitions on a particular disk. If a disk contains multiple disk platters, a partition includes the same portion of each platter.

The first 8K block of usable space on each disk is the *bootblock*. It contains the startup code necessary to *bootstrap*, or start up, the system. The next 8K block contains the *VTOC* (volume table of contents), which describes the partitions found on that disk. The remaining blocks on the disk are grouped into one or more larger pieces called *partitions*.

Each partition can contain a filesystem. Because partitions can be different sizes, a disk can accommodate filesystems of different sizes.

When a filesystem is created on a disk partition, parameters are set to adapt the filesystem to the characteristics of the disk. The parameters include such considerations as the number of blocks per track and the speed at which the platter rotates. This results in more efficient data transfers between the disk and the main computer memory. This in turn affects the speed of the operating system.

Figure 4-2 shows a disk containing the root filesystem on partition *0* and the */usr* filesystem on partition *1*. This is an example only. On your system, these filesystems might be assigned to different partitions, depending on your hardware. The partitions can also be located in different areas of the disk. For example, partition *0* might be the innermost partition, as in this example, or it might be in the middle of the disk.

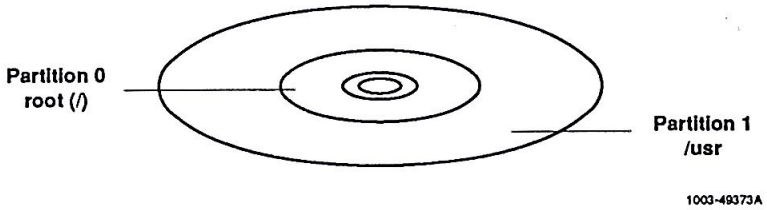


Figure 4-2. Disk partitions containing the *root* and *usr* filesystems.

Cylinders

To understand what happens when a filesystem is created within a partition, you should be familiar with the structure used to store data on a disk.

Partitions are measured in *disk sectors*, which typically contain 512 bytes. A disk platter contains a large number of *tracks*; each track contains a specific number of disk sectors. The read/write head moves in or out to go from track to track when it needs to read data from the disk or write data to the disk.

Data transfers take place in *blocks*, which can vary in size from one filesystem to another. The block size must be a multiple of the sector size (for example, a multiple of 512 bytes). A typical DYNIX/ptx filesystem uses disk blocks containing either 4096 or 8192 bytes. A filesystem using 4096-byte blocks is called a 4K filesystem; a filesystem using 8192-byte blocks is called an 8K filesystem.

Figure 4-3 shows the relationship between the disk sectors, tracks, and blocks on a disk platter. Although the diagram shows only a few tracks, a disk platter typically contains several hundred tracks.

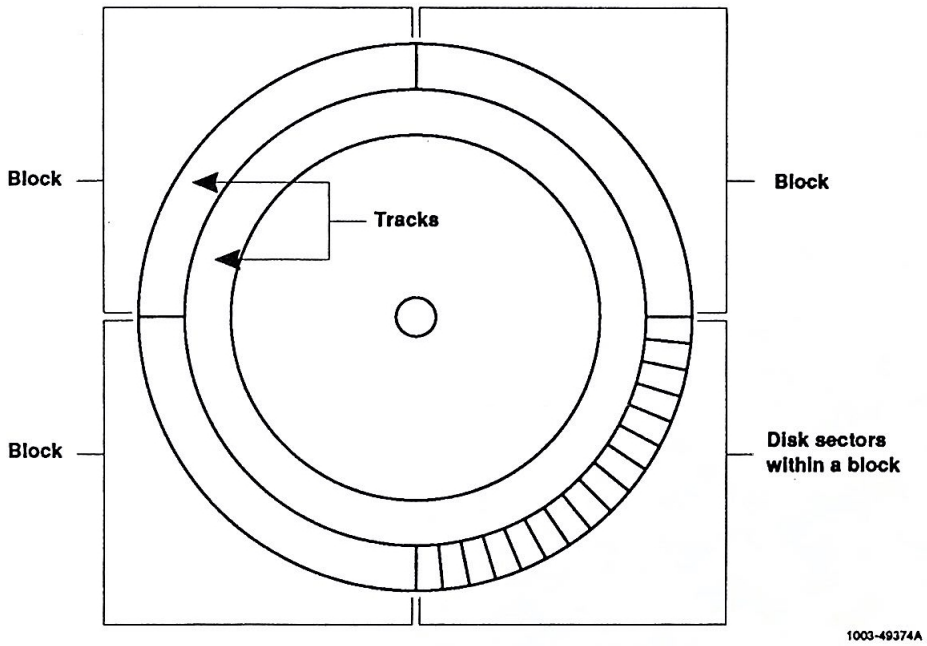


Figure 4-3. The relationship between disk sectors, tracks, and file blocks. In this example, there are 16 sectors per block (8192 bytes per block) and four blocks per track.

The tracks themselves are grouped into *cylinders*; each cylinder contains a specific number of tracks. The read-write head can access all tracks within a cylinder without moving to another position. A cylinder includes the same set of tracks from each usable platter within the disk pack. Figure 4-4 shows a few of the cylinders on a sample disk pack.

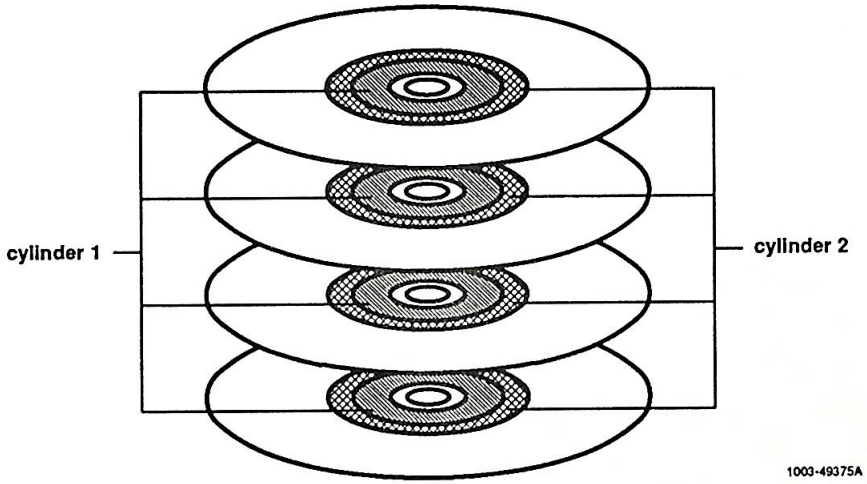


Figure 4-4. Cylinders on a sample disk pack.

Although this diagram shows only two cylinders, a disk can include several hundred cylinders. The type of disk you are using determines the number of cylinders per disk, the number of tracks per cylinder, and the number of sectors per track.

The cylinders within a partition are divided into *cylinder groups*. Each group contains several contiguous cylinders. The filesystem tries to spread its files and directories across the cylinder groups and to store data for a specific file within the same cylinder group. The purpose of doing this is to minimize the number of times the read/write head has to move, resulting in faster data transfers.

4.2.2 The Filesystem Structure

A filesystem contains two special blocks located at the beginning of the partition. The first block is called the *boot block*. If the filesystem is involved in booting the operating system, the block stores booting procedures. If the filesystem is not involved in booting, the boot block is empty.

The next block is called the *superblock*. It contains information about the filesystem, including its name, the number of data blocks it contains, and the maximum number of files it can contain. It also stores disk pack labels and state information about the disk.

The superblock is created when the filesystem is created. Copies of the superblock are kept throughout the filesystem in case the original superblock is lost. Figure 4-5 shows the layout of a sample disk.

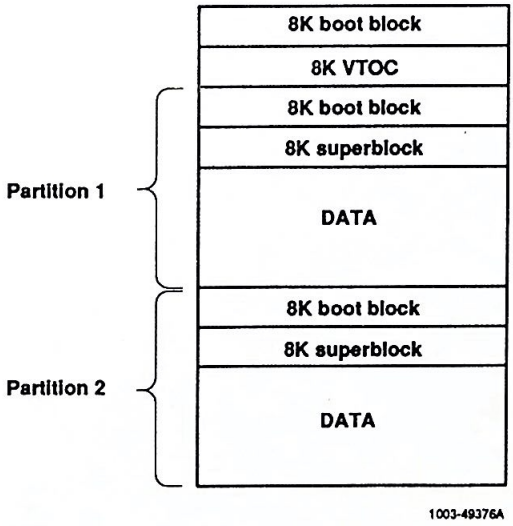


Figure 4-5. Disk layout. Each disk has a boot block and a VTOC. Each partition can contain a filesystem that includes a boot block, a superblock, and data blocks.

Each cylinder group within a filesystem contains data storage blocks, a copy of the superblock, a set of inodes, and information about the free blocks in the cylinder group. When a user creates a file, the operating system uses this information to locate blocks in which to store the contents of the file.

Inodes

Inodes contain information about the files in the filesystem. When a filesystem is created, multiple inodes are created within each cylinder group. Typically, more inodes are created than will be used. When a user creates a file or directory, the filesystem uses an available inode to store the following information about that file or directory.

- The owner of the file
- The owner's group
- The type of file (regular, directory, character, block, FIFO)
- The access permissions for the file
- The size of the file in bytes
- The number of links to the file
- The time the file was last accessed
- The time the file was last modified (written to disk)
- The time the inode itself was last modified
- The location of the disk blocks storing the data in the file

The access time for the file changes whenever the file is read or written to disk. The modification time for the file changes whenever the file is written to disk. The modification time for the inode changes either when the file is modified or when the access permissions, number of links, or file owner changes.

Data Blocks

For a file, data blocks contain the contents of the file. For a directory, data blocks contain the name, inode number, and size of each entry in the directory.

Frequently, the contents of a file are not stored in contiguous data blocks. When the operating system needs to write the contents of a file from disk to memory, the inode information tells the operating system where to find the appropriate data blocks.

Fragments

Many files are not large enough to fill an entire data block. To avoid wasting the remaining space in the block, the filesystem divides each block into *fragments*. The fragment size is specified at the time the filesystem is created. The size must be a multiple of the sector size (for example, a multiple of 512 bytes). It must also evenly divide the number of bytes in a block.

Typically, a fragment is either one-half, one-quarter, or one-eighth of a block. For example, a block containing 8192 bytes could contain either two 4096-byte fragments, four 2048-byte fragments, or eight 1024-byte fragments.

If a file does not require an entire data block for storage, the remaining fragments in the block are available to store the contents of another file.

On an 8K filesystem, if a file uses an indirect pointer (as described in "Indirect Blocks") and contains more than 96 Kbytes, it will not use fragments.

Indirect Blocks

The operating system uses a chain of address blocks to locate the data blocks storing the contents of a file. An inode contains addresses for 15 data blocks. The first 12 addresses (address 0 through 11) point to data storage blocks containing the contents of the file. Only the last direct block pointer in use can point to a fragment.

Address 12 contains a pointer to an *indirect block*. This block stores addresses for additional data storage blocks.

Address 13 contains a pointer to a *double indirect block*. This block contains addresses for a set of indirect blocks, which contain the addresses of data storage blocks.

Address 14 contains a pointer to a *triple indirect block*. This block contains the addresses for a set of double indirect blocks. These blocks point to a set of indirect blocks, which contain addresses for data storage blocks.

Figure 4-6 shows how the addresses in an inode point to the disk storage blocks that actually contain the file.

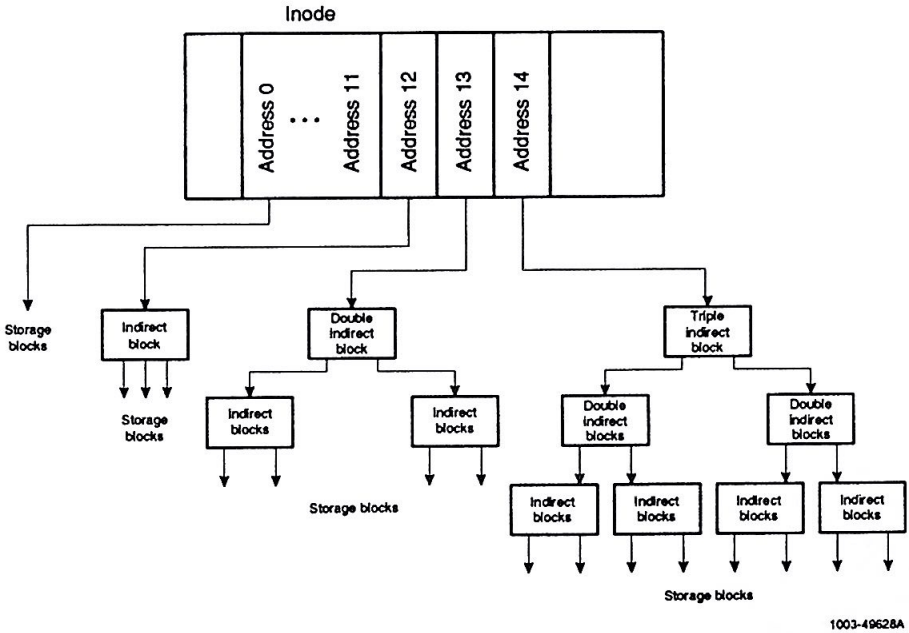


Figure 4-6. Locating disk storage blocks. In most cases, the blocks are filled up in order, starting at the blocks pointed to by address 0. If a file is large enough to fill the blocks pointed to by addresses 0 through 11, it will not use fragments. On an 8K filesystem, it is virtually impossible for a file to fill the triple indirect block.

4.3 Maintaining Filesystems

After a filesystem has been created and made available to users, you should perform the following tasks on a regular basis:

- Monitor disk usage.
- Back up and restore filesystems.
- Check for filesystem consistency.

4.3.1 The Need for Policies

Filesystem administration should be based on establishing a set of policies that are appropriate for your organization. There can be no hard-and-fast rules for such things as the size of filesystems, the number of users in a filesystem, the way in which backups are done, the extent to which users can be allowed to keep inactive files in the system, or the amount of disk space a single user is entitled to occupy. These questions can be resolved only within the context of the organization. The number of users, the type of work they are doing, the number of files needed—all are variables. The responsible administrator must determine what best meets the needs of the organization.

4.3.2 Shell Scripts for Filesystem Administration

Once policies have been agreed on, many of the routine tasks connected with filesystem administration can be incorporated in shell scripts. For example, disk use monitoring can be handled through shell scripts that do the monitoring for you and then transmit messages to the system console when exceptions are detected. Here are a few ideas:

- Use a shell script running under **cron**(1M) control to investigate free blocks and free inodes and to report on filesystems that fall below a given threshold.
- Use a shell script to do automatic cleanups of files that grow.
- Use a shell script to locate excessive disk space usage.

4.3.3 Monitoring Disk Usage

It is important to monitor the disk usage within a filesystem. Following are some areas to watch:

- If the percentage of disk space used is not monitored regularly, it can increase until the allocated space is used up. This causes processes to run very slowly, or not at all. The system spends its time putting out a message about being out of file space.
- There is a natural tendency for users to forget about files they no longer use; the files just sit there taking up space.
- Some files grow larger as a result of normal system use. It is an administrative responsibility to keep them under control.
- Some directories, notably */tmp*, accumulate files during the day. When the system is first brought up, */tmp* needs to have enough free blocks to carry it through to **shutdown**(1M).

The following tasks can help keep disk space uncluttered:

- Monitor the percent of disk space used.
- Monitor files and directories that grow.
- Identify and remove inactive files.
- Identify large space users.

Chapter P4, "Filesystem Administration Procedures," shows how to use the System Administration menus to perform these tasks. This section supplements the procedures and shows some operating system commands that you can use.

4.3.4 Monitoring Percent of Disk Space Used

You can monitor disk space at any time to see how close to capacity your system is running. Until a pattern has emerged, it is advisable to check every day. The following example uses the **df**(1M) command to check disk usage.

```
$ df -t
```

The **-t** option lists total allocated blocks and inodes as well as free blocks and inodes. If you do not specify a filesystem on the command line, the command lists information about all mounted filesystems.

4.3.5 Monitoring Files and Directories that Grow

Almost any system that is used daily has several files and directories that grow through normal use. Following are some examples:

/etc/wtmp	History of system logins
/usr/adm/sulog	History of **su** commands
/usr/lib/cron/log	History of actions of **/etc/cron**
/usr/lib/help/HELPLOG	Actions of **/usr/bin/help**
/usr/lib/spell/spellhist	Words that **spell**(1) fails to match

The frequency with which you should check growing files depends on how active your system is and how critical the disk space problem is. Following is a good technique for keeping files down to a reasonable size. It uses a combination of **tail**(1) and **mv**(1):

```
$ tail -50 /usr/adm/sulog > /tmp/sulog
$ mv /tmp/sulog /usr/adm/sulog
```

This sequence puts the last 50 lines of */usr/adm/sulog* into a temporary file, then moves the temporary file to *usr/adm/sulog*. This step truncates the file to the 50 most recent entries.

4.3.6 Identifying and Removing Inactive Files

Part of the job of cleaning up heavily loaded filesystems involves locating and removing files that have not been used recently. The commands you might use to do this work are described later in this section. Following are some policy decisions for you to consider.

- How long should a file remain unused before it becomes a candidate for removal?
- Should users be warned that old files are about to be purged?
- Should the files be archived or permanently removed?

The **find**(1) command locates files that have not been accessed recently. It searches a directory tree, beginning at a point named on the command line, looking for filenames that match a given set of expressions. When **find** locates a match, it performs a specified action on the file. Following is an example.

```
$ find /usr -type f -mtime +60 -print > /tmp/deadfiles &
```

Here is what the example shows:

/usr Specifies the pathname where **find** is to start. Presumably, your machine is organized in such a way that inactive user files will not often be found in the root filesystem.

-type f Tells **find** to look only for regular files and to ignore special files, directories, and pipes.

-mtime +60
Lists files that have not been modified in the previous 60 days.

-print Tells **find** to print the pathnames of files matching the **-type** and **-mtime** expressions.

> /tmp/deadfiles &
Directs the output to a temporary file and indicates that the process is to run in the background. This is useful if you expect a substantial amount of output.

The **sysadm fileage**(1) command can be used to produce similar information (see Chapter P4).

4.3.7 Identifying Large-Space Users

The following questions can help you determine what the limits for user disk space should be on your system.

- On your system, what constitutes a reasonable amount of disk space for a user?
- If a user exceeds the normal amount by 25%, is it possible that the user's job requires extraordinary amounts of disk space?
- Is your system as a whole running short of space? Do your existing limits need to be reviewed?

Two commands produce useful information in this area: **du**(1) and **find**(1).

du produces a summary of the block counts for files or directories named on the command line. For example, the following command lists the block count for all directories in the */usr* filesystem.

```
$ du /usr
```

Optional arguments allow you to refine the output. For example, **du –s** can be run against each user's login to monitor individual users.

The **find** command can be used to locate specific files that exceed a given size limit.

```
$ find /usr -size +10 -print
```

This example lists the pathnames of all files and directories in the */usr* filesystem that are larger than ten 512-byte blocks. The **sysadm filesize**(1) command can produce similar information (see Chapter P4).

4.4 Filesystem Backup and Restore

The importance of establishing and following a filesystem backup plan is too often not appreciated until information is lost and cannot be recovered. Backing up a filesystem takes time. Trying to recover lost or damaged data from paper records and best-guess-work takes even more time. The value of an effective system backup plan lies in the ability to recover lost or damaged data easily.

System Administration menus are provided to help you with complete and incremental filesystem backups and with restoring backup data. The ability to copy selected directories and files to tape is provided by the **sysadm store** command or the **find**(1) command with **cpio**(1) or **ctccpio**(1M). You can restore directories and files by using **sysadm restore**(1) or the appropriate **cpio** command.

Your backup plan can include any or all of these methods. It is important, however, that you evaluate the need for system backup and form a backup plan. This plan should be reevaluated as the use of your system changes. A System Administration facility for establishing a check of your backup schedule is available; it is described under "Backup Schedule Reminder" later in this section.

4.4.1 Complete Backup

The complete backup provided by the System Administration backup facilities copies the directories and files from a specified mounted filesystem to tape. Complete and incremental backups combine to form a unified backup strategy. Complete backups can be done without intervening incremental backups, but incremental backups must be preceded by at least one complete backup to provide the base.

4.4.2 Incremental Backup

The incremental backup provided by the System Administration backup facilities copies files that have changed since the last backup.

The incremental filesystem backup is fast in comparison to the time required to do a complete backup, because only files that have changed since the last prior backup (whether complete or incremental) are copied. However, while incremental backups save time, they may tie up a sizable number of tapes; restoring from incremental backups can result in your giving back all the time saved.

Not all changed directories and files are copied in an incremental backup. The */etc/save.d/except* file specifies files and directories that are not copied. Figure 4-7 shows a sample */etc/save.d/except* file.

The question of when to do a new complete backup is the critical point of a system backup plan. Here are some things to think about:

- How much time do you want to devote to complete backups? Fewer tapes are required to maintain a backup library if a complete backup is done frequently. A complete backup, however, can take 8 or 10 times as long as any single increment.
- How much time are you willing to devote to restoring a filesystem? All tapes of each incremental backup must be read to restore a filesystem completely. (This is a difficult decision to weigh properly. Ideally, you will never need to restore a whole filesystem from backup.)
- What is the approximate rate of change in the filesystem under scrutiny? Do changes occur throughout the filesystem or in a small percentage of the files? If change is frequent and widespread, it may be best to schedule complete backups more often. If change is concentrated in a few files, perhaps selective backup should be a part of the overall plan.

```
#  Patterns of filenames to be excluded
#  from saving by savefiles.
#  These are ed(1) regular expressions.
/.news_time$
/.yesterday$
/a.out$
/core$
/dead.answer$
/dead.letter$
/ed.hup$
/nohup.out$
/tmp/
^/etc/mnttab$
^/etc/save.d/timestamp/
^/etc/utmp$
^/etc/wtmp$
^/usr/adm/
^/usr/at/
^/usr/crash/
^/usr/dict/
^/usr/games/
^/usr/learn/
^/usr/news/
^/usr/spool/
^/usr/tmp/
```

Figure 4-7. Sample */etc/save.d/except* file.

4.4.3 Backup Schedule Reminder

The File Management Menu contains a backup schedule reminder feature that lets you perform these tasks:

- Schedule reminder messages that say, "If the machine is shut down within this time range, send a message to the console to do a backup of this filesystem."
- Schedule reminder checks that say, "If it gets to be this time and the machine has not been shut down, see if any reminder messages would have been sent had the machine been shut down."

This reminder mechanism does not relieve you of the job of working out a reasonable schedule of backups. It merely nudges you about the schedule you have set up. Chapter P4 shows the steps needed to use the reminder feature. If you prefer, you can set up reminder notices using **cron**(1M) and **crontab**(1).

4.4.4 Restoring a Filesystem from Backup

You can restore individual directories and files or an entire filesystem from system backups. Use the **sysadm restore**(1) command if you need to do this (see Chapter P4 for examples).

To restore a filesystem by using operating system commands, change to the appropriate directory and execute this command:

```
$ cpio -icdumv < device
```

The backup tapes must be restored in the same order in which they were backed up.

4.5 Checking Filesystems

This section describes problems that can corrupt a filesystem and tells how to use the **fsck** program to check for and to repair filesystem problems.

4.5.1 What Can Go Wrong with a Filesystem

Most filesystem corruption problems are caused by hardware failure, program interrupts, or incorrect procedures.

Hardware Failure

There is no effective way to predict when hardware failure will occur. The best way to deal with it is to be sure that recommended diagnostic and maintenance procedures are followed consistently.

Program Interrupts

It is possible that errors causing a program to fail might result in the loss of some data. Programs should be tested exhaustively before they are put into production with valuable data.

Human Error

Probably the greatest cause of filesystem corruption falls under this heading. Following are four rules that can help prevent errors from occurring.

- Always check a filesystem before mounting it. Nothing complicates the problem of cleaning up a corrupt filesystem so much as allowing it to be used when it is bad.
- Always use the **sync** command before shutting down the system and before unmounting a filesystem.
- Never physically write-protect a mounted filesystem, unless it is mounted "read-only."

The random nature of these mishaps simply underscores the importance of establishing and observing good backup practices. It is the most effective form of insurance against data loss.

4.5.2 Checking a Filesystem for Consistency

When the operating system is brought up, a consistency check of the filesystems is done automatically as part of the power-up process. The **fsstat**(1M) command is included as part of that process. **fsstat** returns a code for each filesystem that indicates whether the consistency checking and repair program, **fsck**(1M), should be run.

These same commands, or **sysadm checkfsys**(1), should be used to check filesystems not mounted routinely as part of the power-up process. If inconsistencies are discovered, corrective action must be taken before the filesystems are mounted. The remainder of this section will acquaint you with the **fsck** utility, the type of checking it does in each of its phases, and the repairs it suggests.

Filesystem corruption, while serious, is not all that common. Most of the time a check of the filesystems finds everything all right. The reason we put so much emphasis on filesystem checking is that if errors are allowed to go undetected, the ultimate loss can be substantial.

4.5.3 The fsck Command

The **fsck** command is used to check filesystems. When the operating system is brought up, all filesystems are checked for consistency. This precautionary measure helps to ensure a reliable environment for file storage on disk. The operating system first runs the **fstat** command to check the state of each filesystem. If **fstat** marks a filesystem as "dirty," the system then runs **fsck** to check the filesystem. If **fsck** discovers an inconsistency, corrective action must be taken.

fsck runs in two modes. Typically, the system runs **fsck** noninteractively during a normal boot. When running in this mode, which is called *preening*, **fsck** makes changes to the filesystem only if the changes are known to always be correct.

If **fsck** is checking the root filesystem during a normal boot and finds an unexpected inconsistency, it reboots the system. If this does not fix the problem, **fsck** creates a shell on the console. You should use this shell only to run **fsck** manually to correct the root filesystem. When you exit the shell, the system is rebooted.

---

* The information in section 4.5.3 is taken from the article "Fsck – The UNIX File System Check Program," by Marshall Kirk McKusick and T. J. Kowalski. Revised July 16, 1985. Copyright © 1986, Regents of the University of California.

If, during a normal boot, **fsck** is checking a filesystem other than root and finds an unexpected inconsistency, it exits with a nonzero status, leaving the system running in single-user mode.

Typically, you then run **fsck** manually. When **fsck** runs in this mode (*interactive* mode), it lists each problem and provides a suggested corrective action. You must decide whether **fsck** should make the suggested correction.

Fix Corrupted Filesystems

A filesystem can become corrupted in several ways, most commonly from improper shutdown procedures or hardware failures.

Filesystems can become corrupted during an *unclean halt*. This happens if proper shutdown procedures are not observed or if a mounted filesystem is taken off-line. The most common operator procedural failure is forgetting to **sync** the system before halting the CPU.

Filesystems can become further corrupted if proper start-up procedures are not observed (for example, not checking a filesystem for inconsistencies and not repairing inconsistencies). Allowing a corrupted filesystem to be used (and, thus, to be modified further) can be disastrous.

Any piece of hardware can fail at any time. Failures can be as subtle as a bad block on a disk pack, or as blatant as a nonfunctional disk controller.

Detect and Correct Corruption

Normally, **fsck** is run noninteractively. In this mode, it fixes only those corruptions expected to occur from an unclean halt. These actions are a proper subset of the actions that **fsck** takes when it is running interactively.

This section assumes that **fsck** is running interactively and that all possible errors can be encountered. When **fsck** discovers an inconsistency while running in this mode, it reports the inconsistency and allows you to choose a corrective action.

A quiescent (that is, unmounted and not being written to) filesystem can be checked for structural integrity by performing consistency checks on the redundant data intrinsic to a filesystem. The redundant data is either read from the filesystem or computed from other known values. Because **fsck** is a multipass program, the filesystem must be in a quiescent state when **fsck** is run.

The following subsections discuss methods to discover inconsistencies in the cylinder group blocks, the inodes, the indirect blocks, and the data blocks containing directory entries.

Check the Superblock. The summary information associated with the superblock is the most commonly corrupted item in a filesystem. The summary information is prone to corruption because it is modified with every change to the filesystem blocks or inodes. This information is usually corrupted after an unclean halt.

fsck checks the superblock for inconsistencies involving filesystem size, number of inodes, free-block count, and free-inode count. The filesystem size must be larger than the number of blocks used by the superblock and the number of blocks used by the list of inodes. The filesystem size and layout information are the most critical pieces of information for **fsck**. While there is no way to actually check these sizes, as they are statically determined by **newfs**, **fsck** can check that these sizes are within reasonable bounds. All other filesystem checks require that these sizes be correct. If **fsck** detects corruption in the static parameters of the default superblock, it asks the operator to specify the location of an alternate superblock.

Check Free Blocks. **fsck** checks that blocks marked as free in the cylinder group block maps are not claimed by any files. When all of the blocks have been accounted for initially, **fsck** checks that the number of free blocks plus the number of blocks claimed by the inodes equals the total number of blocks in the filesystem.

If anything is wrong with the block allocation maps, **fsck** rebuilds them based on the list of allocated blocks it has computed.

The summary information associated with the superblock contains a count of the total number of free blocks within the filesystem. **fsck** compares this count to the number of free blocks it found within the filesystem. If the two counts do not agree, **fsck** replaces the incorrect count in the summary information with the actual free-block count.

The summary information counts the total number of free inodes within the filesystem. **fsck** compares this count to the number of free inodes it found within the filesystem. If the two counts do not agree, **fsck** replaces the incorrect count in the summary information with the actual free-inode count.

Check the Inode State. An individual inode is not as likely to be corrupted as the allocation information. However, because of the great number of active inodes, a few inodes are usually corrupted.

fsck checks the list of inodes in the filesystem sequentially, starting with inode 2 and progressing through the last inode in the filesystem. (Inode 0 marks unused inodes; inode 1 is saved for future generations.) It checks the state of each inode for inconsistencies involving format and type, link count, duplicate blocks, bad blocks, and inode size.

Each inode contains a *mode word* that describes the type and state of the inode. Inodes must be one of six types: regular inode, directory inode, symbolic link inode, special block inode, special character inode, or fifo (named pipe). Inodes can be in one of three allocation states: unallocated, allocated, and neither unallocated nor allocated. This last state suggests an incorrectly formatted inode. An inode can get in this state if bad data is written into the inode list. The only possible corrective action for **fsck** is to clear the inode.

Inode Links. Each inode counts the total number of directory entries linked to the inode. **fsck** verifies the link count of each inode by starting at the root of the filesystem and descending through the directory structure. During the descent, **fsck** calculates the actual link count for each inode.

If the stored link count is nonzero and the actual link count is zero, no directory entry appears for the inode. If this happens, **fsck** places the disconnected file in the *lost+found* directory. If the stored and actual link counts are nonzero and unequal, a directory entry may have been added or removed without updating the inode. If this happens, **fsck** replaces the incorrect stored link count with the actual link count.

Each inode contains a list, or pointers to lists (indirect blocks), of all blocks claimed by the inode. Because indirect blocks are owned by an inode, an inconsistency in an indirect block directly affects the inode that owns it.

fsck compares each block number claimed by an inode against a list of already allocated blocks. If another inode already claims a block number, that block number is added to a list of *duplicate blocks*. Otherwise, the list of allocated blocks is updated to include the block number.

If there are any duplicate blocks, **fsck** performs a partial second pass over the inode list to find the inode of the duplicated block. The second pass is necessary because without examining the files associated with these inodes for correct content, not enough information is available to determine which inode is corrupted and should be cleared. If this condition does arise (an unclean shutdown or hardware failure can cause it), the inode with the earliest modify time is usually incorrect and should be cleared. If this happens, **fsck** prompts the operator to clear both inodes. The operator must decide which inode to keep and which inode to clear.

fsck checks the range of each block number claimed by an inode. If the block number is either lower than the first data block in the filesystem or greater than the last data block, the block number is a *bad block number*. Bad blocks in an inode are usually caused by an indirect block that was not written to the filesystem, a condition that can be caused by an unclean shutdown or a hardware failure. If an inode contains bad block numbers, **fsck** prompts the operator to clear the inode.

Inode Data Size. Each inode contains a count of the number of data blocks that it contains. The number of actual data blocks is the sum of the allocated data blocks and the indirect blocks. **fsck** computes the actual number of data blocks and compares that block count against the actual number of blocks the inode claims. If an inode contains an incorrect count, **fsck** prompts the operator to fix it.

Each inode contains a 32-bit size field. The size is the number of data bytes in the file associated with the inode. **fsck** makes a rough check of the consistency of the byte size field. To do this, it computes from the size field the maximum number of blocks that should be associated with the inode and then compares that expected block count against the actual number of blocks the inode claims.

Check the Data Associated with an Inode. An inode can directly or indirectly reference three kinds of data blocks. All referenced blocks must be the same kind. The three types of data blocks are plain data blocks, symbolic link data blocks, and directory data blocks. Plain data blocks contain the information stored in a file; symbolic link data blocks contain the pathname stored in a link. Directory data blocks contain directory entries. **fsck** can check the validity of directory data blocks only.

fsck checks each directory data block for several types of inconsistencies including directory inode numbers pointing to unallocated inodes, directory inode numbers that are greater than the number of inodes in the filesystem, incorrect directory inode numbers for "." and "..", and directories that are not attached to the filesystem. If the inode number in a directory data block references an unallocated inode, **fsck** will remove that directory entry. These conditions can arise from an unclean shutdown or hardware failure, or from bad data being written into a directory data block.

The directory inode number entry for "." must be the first entry in the directory data block. The inode number for "." must reference itself (that is, it must equal the inode number for the directory data block). The directory inode number entry for ".." must be the second entry in the directory data block. Its value must equal the inode number for the parent of the directory

entry (or the inode number of the directory data block if the directory is the root directory). If the directory inode numbers are incorrect, **fsck** will replace them with the correct values. If there are multiple hard links to a directory, the first link encountered is considered the real parent to which ".." should point; **fsck** recommends deletion for the subsequently discovered names.

Filesystem Connectivity. fsck checks the general connectivity of the filesystem. If directories are not linked into the filesystem, **fsck** links the directory back into the filesystem in the *lost+found* directory.

Chapter 5
LP Print Service Administration

5.1 Introduction 5-1
How the LP Print Service Works 5-1
Summary of User Commands 5-2
Summary of Administrative Commands 5-2

5.2 Starting and Stopping the LP Print Service 5-4
Stopping the Print Service Manually 5-4
Starting the Print Service Manually 5-5

5.3 Printer Management 5-6
Defining the Configuration of a Printer 5-6
Printer Name 5-7
Connection Method 5-7
Interface Program 5-9
Printer Type 5-10
Content Types 5-11
Printer Port Characteristics 5-13
Character Sets or Print Wheels 5-14
Forms Allowed 5-18
Printer Faults 5-19
Users Allowed Access to the Printer 5-21
Banner Necessary 5-22
Description 5-22
Default Print Specifications 5-23
Putting It All Together 5-24
Printer Classes 5-24
Setting the Default Destination 5-25
Removing a Printer or Class 5-26
Mounting a Form or Print Wheel 5-26
Accepting Print Requests 5-27
Enabling and Disabling a Printer 5-28
Allowing Users to Enable and Disable Printers 5-29
Displaying Printer Status and Configuration Information 5-29

5.4 Troubleshooting 5-31
No Output – Nothing Prints 5-31

Illegible Output 5-32
Legible Printing, but Wrong Spacing 5-33
Wrong Character Set or Font 5-34
Dial-Out Failures 5-35
Idle Printers 5-35

5.5 Managing the Printing Load 5-37
Moving Requests to Another Printer 5-37
Rejecting Requests for a Printer or Class 5-37
Accepting Requests for a Printer or Class 5-38
Examples 5-38

5.6 Managing Printer Queues 5-40
Printer Queue Setup 5-40
Setting User Priority Limits 5-40
Setting a Default Priority Level 5-41
Examining Priority Limits 5-41
Moving a Request in the Queue 5-41
Changing the Priority for a Request 5-42
Putting a Request on Hold 5-42
Moving a Request to the Head of the Queue 5-43

5.7 Forms Setup 5-44
Defining a Form 5-44
Viewing a Form Definition 5-47
Removing a Form Definition 5-47
Restricting Access to a Form 5-47
Notification to Mount a Form 5-48

Chapter 5
LP Print Service Administration

5.1 Introduction

This chapter tells how to configure and administer the LP print service. Chapter P5 in the first half of this guide provides the procedures you can use to manage the print service.

This chapter includes the following topics:

- Administering the LP print service
- Stopping and starting the LP print service
- Setting up printer configurations
- Managing the printing load
- Setting job priority limits for users
- Managing pre-printed forms

This chapter also describes LP print service files and directories.

5.1.1 How the LP Print Service Works

The LP print service allows you to send a file to be printed while you continue with other work. The LP print service system is software that performs these tasks:

- Receives files sent to the print service.
- Filters the files (if needed) so they can print properly.
- Schedules the work of one or more printers.
- Starts programs that interface with printers.
- Keeps track of the status of jobs.
- Alerts you to printer problems.

- Keeps track of forms currently mounted and alerts you to mount needed forms.
- Issues error messages when problems arise.

5.1.2 Summary of User Commands

The LP print service includes three commands that any user can issue:

cancel(1)	Cancels a print request.
lp(1)	Sends a file or files to a printer.
lpstat(1)	Reports the status of the LP system.

Users can also be given the ability to enable or disable a printer. The idea is that if a user finds a printer is malfunctioning in some way, it should not be necessary to call the administrator to turn the printer off. On the other hand, it may not be reasonable in your printing environment to allow regular users to disable a printer. You can control whether other users have access to the commands to enable or disable a printer.

disable(1)	Deactivates the named printers.
enable(1)	Activates the named printers.

5.1.3 Summary of Administrative Commands

The LP print service includes a separate set of commands for system administration. These commands are found in the */usr/lib* directory. If you expect to use them frequently, you should include that directory in your **PATH** variable. To use the administrative commands, you must be logged in either as **root** or as **lp**, which is a system login.

The administrative commands are as follows:

/usr/lib/accept(1M)
: Permits job requests to be queued for a specified destination.

/usr/lib/reject(1M)
: Prevents jobs from being queued for a specified destination (described on the same manual page as **accept**(1M)).

/usr/lib/lpadmin(1M)
: Sets up or changes printer configurations.

/usr/lib/lpfilter(1M)
: Sets up or changes filter definitions.

/usr/lib/lpforms(1M)
: Sets up or changes definitions for preprinted forms (use **/usr/lib/lpadmin** to mount a form).

/usr/lib/lpmove(1M)
: Moves output requests from one destination to another (described on the same manual page as **lpsched**(1M)).

/usr/lib/lpsched(1M)
: Starts the LP print service.

/usr/lib/lpshut(1M)
: Stops the LP print service (described on the same manual page as **lpsched**(1M)).

/usr/lib/lpusers(1M)
: Sets or changes the default priority level and the priority limits available to users.

The following sections describe these administrative commands in the order in which they are typically used when you set up the LP print service.

5.2 Starting and Stopping the LP Print Service

Under normal operation, you should never have to start or stop the LP print service manually. It starts automatically each time the operating system is started and stops each time the operating system is stopped. However, if you need to stop the LP print service without stopping the operating system as well, follow the procedure described later in this section.

Stopping the LP print service causes all printing to cease within seconds. The print requests that were being printed will be reprinted in their entirety after the LP print service is restarted. The printer configurations, forms, and filters that were in effect when the LP print service was stopped will be restored after the service is restarted.

NOTE

To start and stop the LP print service manually, you must be logged in as either the superuser **root** *or the user* **lp**.

5.2.1 Stopping the Print Service Manually

To stop the LP print service manually, enter the following command:

```
# /usr/lib/lpshut
```

The following message appears and all printing ceases within a few seconds.

```
Print services stopped
```

If you try to stop the LP print service when it is not running, you will see this message:

```
Print services already stopped
```

5.2.2 Starting the Print Service Manually

To restart the LP print service manually, enter the following command:

```
# /usr/lib/lpsched
```

The following message appears:

```
Print services started
```

It might take a minute or two for the printer configurations, forms, and filters to be reestablished. Any saved print requests will then begin printing. If you try to restart the LP print service when it is already running, you will see this message:

```
Print services already active
```

NOTE

The LP print service does not have to be stopped to change printer configurations or to add forms or filters.

5.3 Printer Management

This section tells how to manage printers working with the LP print service. It describes the configuration information you can supply for a printer, tells how to notify the print service when a printer can accept print requests, shows how to enable or disable a printer, and tells how to examine the status of a printer.

5.3.1 Defining the Configuration of a Printer

When you add a printer to the print service, you must define its configuration. The following information can be included in a printer configuration:

- Printer name
- Connection method
- Interface program
- Printer type
- Content types
- Printer port characteristics
- Character sets or print wheels
- Forms allowed
- Fault alerting
- Fault recovery
- Use restrictions
- Banner necessary
- Description
- Default printing attributes

You don't need to supply all of this information. Only the printer name and the connection method are required. The more information you provide, however, the better the printer will be managed for you and the better it will be represented to the people using the LP print service.

The descriptions in the following sections will help you understand what the printer configuration information means and how it is used. You can then decide how to configure your printers. While you can follow each of the sections in order and correctly configure a printer in several steps, you might want to wait until you've read all of the sections and then add the printer in a single step.

Printer Name

The printer name identifies the printer. You will use this name when you want to change the printer configuration or manage the printer. The users on your system will use the name to print a file on that printer.

A printer name can be up to 14 characters long. It can include numbers, letters, and underscores. You can select any name you like, but it is helpful to choose descriptive names. For example, you could name a laser printer *laser*. If you have several laser printers, you could name them *laser1*, *laser2*, and so on.

Don't try to fit a lot of descriptive information into the name; there is a better place for this information (see "Description," later in this section). The name doesn't need to identify the type of printer—people who need to use a particular type of printer can specify it by type, not name (see "Printer Type," later in this chapter).

To specify a printer name, use a command in the following form. (Don't do it yet; you also need to specify the connection method as described in the next section.)

```
# /usr/lib/lpadmin -pprinter-name
```

The LP service does not provide default names; you must name every printer.

Connection Method

The LP print service allows you to connect printers in a variety of ways. You can connect printers directly to the computer or you can connect them through a network or a dialed modem, where they can be shared with other computers or workstations. After the printer is connected to your system, you should describe the connection method to the LP print service. The following sections describe the commands required to do this.

Adding a Directly Connected Printer. A direct connection is the default value. If you used this method to connect your printer to your computer, type a command in the following form to add the printer to the print service:

/usr/lib/lpadmin –p *printer-name* **–v** *pathname*

pathname is the name of the special file representing the printer port.

Adding a Printer to be Used as a Login Terminal. Some directly connected printers can also be used as terminals for login sessions. If you want your printer to be used as a login terminal, type a command in the following form to add the printer to the LP print service:

/usr/lib/lpadmin –p *printer-name* **–v** *pathname* **–l**

pathname is the name of the special file representing the printer port. The **–l** option indicates that the printer should be automatically disabled when the LP print service starts up. This allows users to log in on the printer. You will need to enable the printer to allow it to be used for printing.

Adding a Printer Connected Through a Modem or Network. There are two ways to make nondirect connections: through a dial-up modem or over a network. The LP print service uses the Basic Networking Utilities to handle both methods of nondirect connection. If a dial-out modem is used, three prerequisites must be satisfied: the printer must be connected via a dialed modem, a dial-out modem must be connected to the computer, and Basic Networking must know about this modem.

Printers connected over any other type of network require that a "system name" be given for each printer. This is the name of an entry in the *Systems* file or a related file. Although the printer is not a system, the *Systems* file can still be used to record the access method (no login information will be given, of course).

Because the **cu** program accesses a printer in the same way the LP print service does, you should set up the files as though preparing access to the printer for **cu**. The **cu** command is not used to access printers, but can serve as a test when setting up files. If **cu** can access a printer, the LP print service will be able to access it, too. (See Chapter 7, "Basic Networking," for details about setting up network connections.)

To add a printer that is connected over a modem or network, type a command in this form:

/usr/lib/lpadmin –p *printer-name* **–U** *dial-info*

dial-info is either the telephone number to be dialed to reach the printer's modem or the system name entered for the printer in the Basic Networking *Systems* file.

You must include either the **–U** or **–v** option on the **lpadmin** command line. If the printer is to be used as a login terminal, you must also include the **–l** option.

A note on dial-out or network printers: if the printer or port is busy, the LP print service will automatically retry later. If the printer is busy, the retry rate is 10 minutes; if the port is busy, the rate is 20 minutes. These rates are not adjustable. However, you can force an immediate retry by issuing an **enable** command for the printer. If the port or printer is likely to be busy for an extended period, you should issue a **disable** command.

The **lpstat –p** command reports the reason for a failed dial attempt. Also, if you are alerted to a dialing fault, the alert message will give the reason for the fault. These messages are identical to the error messages produced by the Basic Networking Utilities for similar problems.

Interface Program

The LP print service uses an interface program to manage the printer each time a file is printed. An interface program has four main tasks:

- To initialize the printer port (the connection between the computer and the printer)
- To initialize the printer (restore it to a normal state in case a previously printed file has left it in an unusual state) and set the character pitch, line pitch, page size, and character set requested by the user
- To print a banner page
- To run a filter to print the file

If you do not choose an interface program, the standard program provided with the LP print service will be used. This should be sufficient for most of your printing needs. You can modify the default program or write your own interface program and then specify it when you add a new printer. (See "Customizing the Print Service," later in this chapter, for more information.)

If you will be using the standard interface program, you don't need to specify it when adding a printer. If you will be using a different interface program, you can refer to it either by specifying its full pathname or by referring to another printer using that program.

NOTE

You can also specify an interface program by naming a model interface program. The models provided with the old LP Spooling Utilities have been carried over, but they do not support all the new features. These models will be phased out in a future release.

To identify a customized interface program by name, specify the printer name and the pathname of the interface program as follows:

/usr/lib/lpadmin -p *printer-name* **-i** *pathname*

To identify a customized interface program by reference to another printer, type a command in this form:

/usr/lib/lpadmin -p *printer-name1* **-e** *printer-name2*

Replace *printer-name1* with the name of the printer you are adding; replace *printer-name2* with the name of the printer that is currently using the customized interface program.

To identify an interface program by reference to a model interface program, type a command in this form:

/usr/lib/lpadmin -p *printer-name* **-m** *model-name*

Printer Type

The LP print service uses the printer type to extract information about the printer from the Terminfo database. The LP print service will then warn you if some of the configuration information you provide isn't appropriate for the printer. The information in the Terminfo database also describes the control data used to initialize the printer before printing a file. While you are not required to specify a printer type, you are urged to specify one so that better print services will be provided.

The printer type is the generic name for the printer. Typically, it is derived from the manufacturer's name.

Specify the printer type as follows:

```
# /usr/lib/lpadmin -p printer-name -T printer-type
```

If you do not define the printer type, the default value `unknown` will be used. This will produce empty results when the LP print service looks up information about the printer. The print service will not be able to verify certain requests or to initialize the printer.

Content Types

The content type information tells the LP print service the types of files that can be printed. Most printers can print only one type of file. For them, the content type is likely to be identical to the printer type. Some printers, though, can accept several different types of files and print their contents properly. When adding this kind of printer, you should list the names of the content types it accepts.

When a user submits a print request, the print service searches for a printer capable of handling the job. The print service can identify an appropriate printer through either the content type name or the printer type name. Users can specify either a content type or printer type name (or no name) when issuing a print request.

Select content type names that are meaningful to you and to the people using the printer. A content type name can be up to 14 characters long and can include only letters, digits, and underscores. (The names `simple`, `terminfo`, and `any` have specific meanings to the LP print service; be sure to use them consistently.)

If several printers can print the same content type, specify the same content type names when you add those printers. Users can then specify the same name to identify the type of file they want printed regardless of the printing destination.

For example, several manufacturers produce printers that accept ANSI X3.64 defined escape sequences. However, the printers might not support all of the ANSI capabilities or might support different sets of capabilities. To differentiate among these printers, you could specify different content type names for them.

You do not have to list the content types for a printer. If you don't, the printer type will be used as the name of the content type the printer can handle. If you have not specified a printer type, the LP print service will assume that the printer can print only files of content type `simple`. This may be sufficient if you will require people to pick the proper printer and to make sure the files are properly prepared for the printer.

One type of file often encountered is called `simple`. This file is assumed to contain printable ASCII characters and the following control characters:

BackSpace	Moves the carriage back one space, except at the beginning of a line.
Tab	Moves the carriage to the next tab stop; by default, tabs are spaced every 8 columns on most printers.
LineFeed	Moves the carriage to the beginning of the next line (may require special port settings for some printers; see "Printer Port Characteristics").
FormFeed	Moves the carriage to the beginning of the next page.
CarriageReturn	Moves the carriage to the beginning of the same line (may fail on some printers).

If a printer can handle a `simple` type of file, you should include it in the content type list when you add the printer. If you don't want a printer to accept files of type `simple`, you must specify an alternate list of content types the printer can accept. (The printer type is a good name to use if no other type is appropriate.)

The names `terminfo` and `any` should not be used as content types when adding a printer. The name `terminfo` doesn't refer to a particular type of file, but instead refers to all of the types represented in the Terminfo database. It is not likely that any printer is capable of handling all the types listed in the database. However, this name is reserved for describing possible filter capabilities. The content type `any` is reserved for describing the types of files a filter can accept or produce.

Specify the list of content types as follows:

```
# /usr/lib/lpadmin -p printer-name -I content-type-list
```

The *content-type-list* is a list of names separated by either commas or spaces. If you use spaces to separate the names, enclose the entire list (but not the **-I**) in quotes. If you do not define the types of files a printer can accept, the

LP print service will assume it can take type `simple` and a type with the same name as the printer type (if the printer type is defined).

Printer Port Characteristics

The interface program must set printer port characteristics for directly connected printers and for printers connected over some networks. These characteristics define the low-level communications with the printer and include the baud rate, use of XON/XOFF flow control, 7, 8, or other bits per byte, style of parity, and output post-processing. The standard interface program uses the **stty** command to initialize the printer port. It sets the baud rate and a few other default characteristics, as described below:

Default	Meaning
`9600`	9600 baud rate
`cs8`	8-bit bytes
`-cstopb`	1 stop bit per byte
`-parenb`	No parity generation
`ixon`	Enable XON/XOFF flow control
`-ixany`	Allow only XON to restart output
`opost`	Post-process data stream
`-oluc`	Don't map lowercase to uppercase
`onlcr`	Map linefeed into carriage-return/linefeed
`-ocrnl`	Don't map carriage-return into linefeed
`-nocr`	Output carriage-returns even at column 0
`nl0`	No delay after linefeeds
`cr0`	No delay after carriage-returns
`tab0`	No delay after tabs
`bs0`	No delay after backspaces
`vt0`	No delay after vertical tabs
`ff0`	No delay after form-feeds

The default characteristics might be sufficient for your printers. However, printers vary enough that you might need to set different characteristics. See the description of the **stty**(1) command in the *Reference Manual* to find the complete list of characteristics.

If you have a printer that requires printer port characteristics other than those handled by the **stty** program, you will have to customize the interface program.

When you add a new printer, you can specify an additional list of port characteristics that should be applied when printing a file. Because your list will be applied after the default list, your list doesn't need to include the default items you don't want to change. Specify the additional list as follows:

```
# /usr/lib/lpadmin -p printer-name -o "stty='stty-option-list'"
```

If you include more than one item in the *stty-option-list*, both double quotes and single quotes are needed. If you do not include alternate printer port characteristics, the default list shown earlier will be used.

As an example, suppose your printer is to be used for printing graphical data where linefeed characters should be output alone, without an added carriage-return. You would enter the following command:

```
# /usr/lib/lpadmin -p printer-name -o "stty=-onlcr"
```

The single quotes are omitted because the list contains only one item.

As another example, suppose your printer requires odd parity for data sent to it. You would enter the following command:

```
# /usr/lib/lpadmin -p printer-name -o "stty='parenb parodd cs7'"
```

Character Sets or Print Wheels

Many printers can print in different font styles. Some printers use print wheels to do this, others use font cartridges or preprogrammed, selectable character sets. When you add a printer, you can specify the print wheels, font cartridges, or character sets available with the printer. Only one of these is assumed to apply to each printer.

To the LP print service, print wheels and changeable font cartridges are the same because they require you to intervene and mount a new print wheel or font cartridge. This section discusses only print wheels and character sets. However, the print wheel information also applies to font cartridges.

When you list the print wheels or character sets available on a printer, you must assign names to them. Because different printers can have similar print wheels or character sets, you should use common names for all printers. This allows a user to submit a file for printing and to ask for a particular font style, without regard for which printer will be used or whether a print wheel or selectable character set is used.

If a printer has mountable print wheels, you need to list only their names. Use a command in this form:

```
# /usr/lib/lpadmin -p printer-name -S printwheel-list
```

The *printwheel-list* is a list of names separated by commas or spaces. If you use spaces to separate the names, enclose the entire list (but not the **-S**) in quotes.

If a printer has selectable character sets, you need to list their names and to map each character set into a name or number that identifies it in the Terminfo database. You can use the following command to determine the names of the character sets listed in the Terminfo database. *printer-type* is the name of the printer type in question.

```
$ TERM=printer-type tput csnm 0
```

The name of the character set obtained by default after the printer is initialized should appear. Repeat the command, using 1, 2, 3, and so on (in place of the 0), to see the names of the other character sets. In general, the Terminfo names should closely match the names used in the printer documentation. However, because not all manufacturers use the same names, the Terminfo names may differ from one printer type to the next.

NOTE

For the LP print service to be able to find the names in the Terminfo database, you must specify a printer type. For more information, see "Printer Type," earlier in this chapter.

To specify a list of character set names and to map them into Terminfo names or numbers, enter the following command:

```
# /usr/lib/lpadmin -p printer-name -S character-set-list
```

The *character-set-list* is a list of names separated by commas or spaces. Each item in the list looks like one of the following:

```
csN=character-set-name
character-set-name1=character-set-name2
```

The *N* is a number from 0 to 63 that identifies the number of the character set in the Terminfo database. The *character-set-name1* identifies the character set by its Terminfo name. In either case, the value to the right of the equals sign (=) is the name you want to use as an alias for the character set.

NOTE

You do not have to provide a list of aliases for the character sets if the Terminfo names are adequate. You can refer to a character set by number, by Terminfo name, or by your alias.

For example, suppose your printer has two selectable character sets (set #1 and #2) in addition to the standard character set (set #0). The printer type is `5310`. You would enter the following commands to determine the names of the selectable character sets.

```
TERM=5310 tput csnm 1
english
TERM=5310 tput csnm 2
finnish
```

The words `english` and `finnish` are the names of the selectable character sets. You feel that the name `finnish` is adequate for referring to character set #2, but better names are needed for the standard set and set #1. You could enter the following command to define synonyms.

```
# /usr/lib/lpadmin -p printer-name -S "cs0=american, english=british"
```

If you do not list the print wheels or character sets that can be used with a printer, the LP print service will assume the following:

- A printer that takes print wheels has only a single, fixed print wheel and people cannot ask for a special print wheel when using the printer.
- A printer that has selectable character sets can take any `cs`*N* name or Terminfo name known for the printer.

Alerts to Mount Print Wheels or Font Cartridges. When a print request requires a specific print wheel or font cartridge, the request remains in the queue until you tell the print service that you have mounted the print wheel or font cartridge.

The print service can notify you when a print request requires a particular print wheel or font cartridge. The **–A** option to the **lpadmin** command lets you specify how you want to be alerted: by a mail message, by a message written on your terminal, or by another command. You can also choose not to receive alerts.

In the following commands, the **–W** *min* option indicates the number of minutes between alerts. If you want to receive only one alert, omit this option. The **–Q** *integer* option lets you specify the number of requests that must be waiting before you are alerted. If you do not specify an alert method, you will need to check for print requests requiring a specific print wheel or font cartridge.

- To receive a mail message (via the **mail** program), type a command in this form:

  # **lpadmin –S** *printwheel/cartridge* **–A mail –Q** *integer* **–W** *min*

- To have a message written to your terminal (via the **write** command), type a command in this form:

  # **lpadmin –S** *printwheel/cartridge* **–A write –Q** *integer* **–W** *min*

- To specify another command to be run when a print request requires a specific print wheel or font cartridge, specify a command in this form:

  # **lpadmin –S** *printwheel/cartridge* **–A** *'command'* **–Q** *integer* **–W** *min*

  The shell environment in effect when you issue this command is saved and then restored when *command* executes. This includes the environment variables, user and group IDs, and the current directory.

  If you want another person to be notified when a print wheel or font cartridge needs to be mounted, follow the **–A** option with the person's login name.

  –A 'mail *user'*

  –A 'write *user'*

- If you don't want to receive an alert, type a command in this form:

  # **/usr/lib/lpadmin –S** *printwheel/charset* **–A none**

If you type **all** as the name of the print wheel or character set, the notification method will apply to all print wheels and character sets for which you have defined an alert method.

If you are receiving repeated alerts, you can type the following command to tell the print service to stop sending alerts.

/usr/lib/lpadmin –S *printwheel/charset* **–A quiet**

Forms Allowed

If you are using preprinted forms, you can specify the forms that can print on a specific printer. You might want to do this if some forms are the wrong size for a printer or the print quality is not adequate for certain forms. (The forms must be set up in the print service; see "Forms," later in this chapter, for more information.)

If you don't set up this option, the print service assumes that all forms are denied for the printer. You can still mount any form on the printer. However, you will see a warning message when you notify the print service that you are mounting a form.

You can use an allow list and a deny list to specify the forms each printer can handle. The allow list contains forms the printer can print; the deny list contains forms that cannot be used. If there are entries in the allow list, the print service ignores the deny list. If there are no entries in the allow list, the print service uses the deny list. If both lists are empty, all forms can be used on the printer.

To add forms to the allow list, type a command as shown below. If the form name was already in the deny list, it will be removed from that list. To allow all forms, specify **allow:all**.

```
# lpadmin -p printer -f allow:forms
```

To add forms to the deny list, type a command as shown below. If the form name was already in the allow list, it will be removed from that list. To deny all forms, specify **deny:all**.

```
# lpadmin -p printer -f deny:forms
```

If you specify multiple forms, use spaces or commas to separate the names. If you use spaces, enclose the list (including **allow:** or **deny:**) in quotes.

You will receive a warning message if you mount a form that is denied for the printer or that doesn't match the capabilities of the printer. If a user requests a denied form, the print service will refuse the request unless the form is already mounted.

Printer Faults

The LP print service can detect printer faults ranging from a need to replace toner to a local power failure or printer failure. The print service recognizes two types of printer fault indicators: a drop in carrier (the signal indicating the printer is on line) and an XOFF that is not followed by an XON. However, you can add filters that can recognize other printer fault indicators.

Fault Alert Notification. The print service can notify you when it detects a fault. The **–A** option to the **/usr/lib/lpadmin** command lets you specify how you want to alerted: by a mail message, by a message on your screen, or by another command. You can also choose not to receive alerts. The **–W** option indicates the number of minutes between alerts. If you want to receive only one alert, omit this option. If you do not specify an alert method, you will receive a single mail message when a printer fault occurs.

- To receive a mail message (via the **mail** program), type a command in this form:

  # **/usr/lib/lpadmin –p** *printer* **–A mail –W** *minutes*

- To have a message written to your terminal (via the **write** command), type a command in this form:

  # **/usr/lib/lpadmin –p** *printer* **–A write –W** *minutes*

- To specify another command to be run when a fault condition exists, type a command in this form:

  # **/usr/lib/lpadmin –p** *printer* **–A** *'command'* **–W** *minutes*

  The shell environment in effect when you enter this command is saved and then restored when *command* executes.

  If you want another person to be notified when a printer fault occurs, follow the **–A** option with the appropriate login name.

  –A 'mail *user'*

  –A 'write *user'*

- If you don't want to be alerted when a printer fault occurs, type a command in this form:

  # **/usr/lib/lpadmin –p** *printer* **–A none**

  In this case, you must locate faults yourself. The LP print service will not use a printer that has a fault.

If you type **all** as the *printer* in any of these commands, the alerting condition applies to all printers.

NOTE

If you have not added a filter that provides better fault detection, the LP print service cannot automatically determine when a fault has been cleared. It simply tries to print another file and assumes the fault is cleared when it can print the file. If you asked for only one alert per fault, you will not receive another alert even if the fault is fixed and the printer faults again before the system can print a file. To prevent this situation, require repeated alerts or select the **-F wait** *option for fault recovery (see "Fault Recovery," following).*

If you are receiving repeated alerts for a printer fault, you can issue the following command to tell the LP print service to stop sending alerts for that fault.

```
# /usr/lib/lpadmin -p printer-name -A quiet
```

Fault Recovery. The LP print service recovers from a printer fault in one of these ways:

- It continues printing at the top of the page where printing stopped.
- It reprints the entire print request that was active when the fault occurred.
- It waits for you to reenable the printer.

The print service can resume printing at the top of the page where printing stopped only if you are using a filter that can wait for a printer fault to be cleared. Such a filter probably needs detailed knowledge of the control sequences used by the printer so that it can keep track of page boundaries and know where printing stopped. The default filter used by the LP print service cannot do this. The print service will notify you if recovery cannot proceed as you want.

To tell the print service to resume printing at the top of the page, type a command in this form:

lpadmin -p *printer* **-F continue**

To tell the print service to reprint the entire request, type a command in this form:

lpadmin -p *printer* **-F beginning**

To tell the print service to wait until you enable the printer, type a command in this form:

lpadmin -p *printer-name* **-F wait**

If you don't set up this option, the print service will resume printing on the page where printing stopped. If that is not possible, it will reprint the entire request.

If you specify `continue` as the recovery method but the interface program cannot detect when the printer fault has been cleared, printing will be attempted every few minutes until it succeeds. To force the LP print service to retry immediately, enable the printer.

Users Allowed Access to the Printer

You can limit the use of a printer to specific users. If you don't set up this option, everybody can use the printer.

You can use an allow list and a deny list to specify the users able to access the printer. The allow list contains names of users allowed to use the printer; the deny list contains users that cannot use the printer. If there are entries in the allow list, the print service ignores the deny list. If there are no entries in the allow list, the print service uses the deny list. If both lists are empty, everyone can use the printer.

To add user names to the allow list, type a command as shown below. If a user name was already in the deny list, it will be removed from that list. To allow all users, specify **allow:all**.

lpadmin -p *printer* **-u allow:***users*

To add user names to the deny list, type a command as shown below. If a user name was already in the allow list, it will be removed from that list. To deny all users, specify **deny:all**. In this case, only the user **lp** and the superuser (**root**) can use the printer.

lpadmin -p *printer* **-u deny:***users*

If you specify multiple user names, use commas or spaces to separate the names. If you use spaces, enclose the list (including **allow:** or **deny:**) in quotes.

Banner Necessary

A banner page precedes each print job. It shows who requested the print job, the request ID for the job, and when the output was printed. It also allows for an optional title that the requester can use to better identify a printout. The banner page also makes it easier to separate print jobs for different users.

Sometimes a user needs to avoid printing a banner page. For example, a banner page is unnecessary when printing forms such as invoices or checks.

Enter the following command to allow a user to skip the banner page:

```
# /usr/lib/lpadmin -p printer-name -o nobanner
```

To reverse this choice, enter the following command.

```
# /usr/lib/lpadmin -p printer-name -o banner
```

If you do not allow users to skip the banner page, the LP print service will reject all attempts to avoid printing a banner page. This is the default action.

Description

One way to give users helpful information about a printer is to add a description to the print service. The description can contain any message you like, such as the location of the printer or the contact for printer problems.

This message appears when users issue the **lpstat -D -p** *printer-name* command.

To add a description to a printer configuration, enter a command in the following form.

```
# /usr/lib/lpadmin -p printer-name -D 'text'
```

The *text* is the message. If the message contains blanks or other characters that the shell might interpret, enclose the message in quotes. The LP print service does not provide a default description.

Default Print Specifications

A print request can include the page size, character pitch, and line pitch to be used for that job. If a print request doesn't include these specifications, the default values are used.

For each printer, you can specify the default page size and print spacing. For example, one printer might produce wide (132-column) output, another might produce 80-column by 66-line output, and another might produce letter quality (12 characters per inch, 8 lines per inch) output. If you set up these values, users can request an output style and the print service will send the requests to the appropriate printer.

You can specify four default settings: page width, page length, character pitch, and line pitch. You can specify the page width as either columns, inches, or centimeters; the page length can be either lines, inches, or centimeters. The character pitch and line pitch can be specified as the number of characters and lines either per inch or per centimeter. The character pitch can also be specified as **pica** for 10 characters per inch (cpi), **elite** for 12 cpi, or **compressed** for the maximum cpi the printer can provide (up to a limit of 30 cpi).

The following commands allow you to set default printing attributes:

```
# lpadmin -p printer -o width=number

# lpadmin -p printer -o length=number

# lpadmin -p printer -o cpi=number

# lpadmin -p printer -o lpi=number
```

To indicate inches, add the letter **i** to the *number*. To indicate centimeters, add the letter **c**. It is unnecessary to add **i** for the character pitch (cpi) or line pitch (lpi). You can also specify **pica**, **elite**, or **compressed** as the character pitch.

If you don't provide defaults, the print service uses the print specifications available when the printer was initialized. The following command reports the default page size and print spacing.

```
$ lpstat -p printer -l
```

These defaults are calculated from the Terminfo database entry for the printer. (For this to work, you must specify a printer type as described earlier in this chapter.)

Putting It All Together

It is possible to add a new printer by completing a number of separate steps, as shown in the previous discussions. You might find it easier, however, to combine all of the necessary arguments in one or two commands. Following are some examples.

Example 1. Add a new printer called *lp1* on port */dev/ttyGA/GAAH*. It should use the standard interface program, have a default page size of 90 columns by 71 lines, and linefeeds should not be mapped into carriage return/linefeed pairs. (This example is split into two lines for readability.)

```
# /usr/lib/lpadmin -p lp1 -v /dev/ttyGA/GAAH -T 455 -o "width=90 length=71
stty=-onlcr"
```

Example 2. Add a new printer called *laser* on port */dev/ttyGA/GAAI*. It should use a customized interface program, be able to handle three file types (`i10`, `i300`, and `impress`), and be available for use only by the users *doceng* and *docpub*. (This example is split into two lines for readability.)

```
# /usr/lib/lpadmin -p laser -v /dev/ttyGA/GAAI -i /usr/doceng/laser_intface
-I "i10,i300,impress"-u "allow:doceng,docpub"
```

Example 3. When adding the *lp1* printer in the first example, we forgot to set the alerting. We'll have the LP print service alert us every 10 minutes after a fault until we fix the problem.

```
# /usr/lib/lpadmin -p lp1 -A write -W 10
```

5.3.2 Printer Classes

The LP print service allows you to treat a collection of printers as a single class. When issuing a print command, users can specify the class of printers that is to print the request. The LP print service then sends the print request to the first available printer in the class. This allows faster turnaround, as printers are kept as busy as possible.

If you allow users to submit print requests for a particular type of file, it is unnecessary to set up printer classes. The **lp -T** *type* command allows a user to specify the type of file to be printed. The print job will be sent to the first available printer that can print that type of file. The LP print service avoids using a filter. If possible, it chooses a printer that can print the file directly, rather than a printer that would require the file to go through a filter.

Classes do have uses, however. For example, a class can include a set of printers that should be used in a particular order. If you have a high-speed printer and a low-speed printer, you might want the high-speed printer to handle as many print requests as possible, reserving the low-speed printer for times when the other printer is busy. The LP print service always checks for an available printer in the order the printers were added to a class. If you added the high-speed printer to the class before you added the low-speed printer, the LP print service will route print requests to the high-speed printer first.

If you want to add a printer to a class, use a command in this form:

```
# /usr/lib/lpadmin -p printer-name -c class-name
```

If the class (*class-name*) doesn't exist, it will be created.

You cannot use the same name for a class and a printer.

5.3.3 Setting the Default Destination

The LP print service allows you to specify a default destination, which can be either a single printer or a class of printers. If a user does not specify a destination on the command line and has not set the **LPDEST** environment variable, the print request goes to the default destination.

To set up a default destination, type a command in the following form. The printer or class you specify must already exist.

```
# /usr/lib/lpadmin -d printer-or-class-name
```

If you want to remove the default destination, omit the *printer-or-class-name*, as in the following example.

```
# /usr/lib/lpadmin -d
```

If you don't set a default destination, users must either specify a destination on the command line or set the **LPDEST** environment variable to the name of a printer or class.

5.3.4 Removing a Printer or Class

You can remove a printer or class from the LP print service if it has no pending print requests. If there are pending requests, you must first either move them to another printer or class (by using the **lpmove** command), or delete them (by using the **cancel** command).

Removing the last remaining printer from a class automatically removes the class as well. However, the removal of a class does not cause the removal of printers that were members of the class. If the printer or class that you remove is also the system default destination, the system will no longer have a default destination.

To remove a printer or class, enter a command in this form:

```
# /usr/lib/lpadmin -x printer-or-class-name
```

If you want to remove a printer from a class without deleting the printer, enter a command in this form:

```
# /usr/lib/lpadmin -p printer-name -r class-name
```

5.3.5 Mounting a Form or Print Wheel

If a print request requires a specific form or print wheel, you must mount the form or print wheel before the job can print. If alerting has been set on the form or print wheel, you will be alerted when the number of print requests waiting for the form or print wheel reaches the number you specified.

Mounting a form or printwheel involves loading it onto the printer and then telling the LP print service that it is mounted. Because the LP print service will continue to use the printer to print other files, you will probably have to disable the printer first. The proper procedure is as follows:

1. Use the **disable** command to disable the printer.
2. Mount the new form or printwheel as described in the following paragraphs.
3. Use the **enable** command to reenable the printer.

The **disable** and **enable** commands are described in "Enabling and Disabling a Printer," later in this chapter.

After mounting a print wheel, enter a command in the following form to tell the LP print service where the print wheel is mounted.

/usr/lib/lpadmin –p *printer-name* **–M –S** *printwheel-name*

After loading a form, enter a command in the following form to tell the LP print service where the form is loaded.

/usr/lib/lpadmin –p *printer-name* **–M –f** *form-name* **–a –o filebreak**

When you mount a form, you might want to make sure it is lined up properly. If an alignment pattern has been registered with the form, you will be asked to press Return before a copy of the alignment pattern prints. After the pattern prints, you can adjust the printer and press Return again. If no alignment pattern has been registered, you won't be asked to press Return. You can omit the **–a** and **–o filebreak** options if you won't be using the alignment pattern.

The **–o filebreak** option tells the LP print service to add a formfeed after each copy of the alignment pattern. The actual control sequence used for the formfeed depends on the printer involved and is obtained from the Terminfo database. If the alignment pattern already includes a formfeed, omit the **–o filebreak** option.

When you are ready to unmount a form, use a command in this form:

/usr/lib/lpadmin –p *printer_name* **–M –f none**

If you are unmounting a print wheel, enter a command in this form:

/usr/lib/lpadmin –p *printer_name* **–M –S none**

5.3.6 Accepting Print Requests

After you've defined a printer configuration, you must tell the LP print service to accept print requests for the new printer. You will also need to enable the printer.

The **accept** command tells the LP print service to accept print requests for the new printer. This command is described more fully in a later section, "Managing the Printing Load." For now, simply type a command in the following form:

/usr/lib/accept *printer-or-class-name*

This command is also used to tell the LP print service to start accepting print requests for a class.

5.3.7 Enabling and Disabling a Printer

After telling the LP print service to accept print requests for a printer, you must enable the printer to allow the requests to print. (You do not need to enable classes.) Use the **enable** command to enable a printer.

```
# enable printer-name
```

You can specify several printers on the command line. Use spaces to separate the printer names.

Occasionally, you will need to disable a printer. For example, this should be done before you change a form or print wheel or whenever you want to interrupt a print request. Disabling a printer stops further print requests from being printed. However, it does not stop the LP print service from accepting new print requests for the printer.

Disabling a printer also stops the request currently being printed. The request is then returned to the queue so it can be printed later. However, you can have the LP print service cancel the request or wait until the current request finishes and then disable the printer.

To disable the printer, stop the request being printed, and return it to the queue to be printed later, type a command in this form.

```
# disable -r "reason" printer-name
```

To disable the printer after the current request finishes printing, type a command in this form:

```
# disable -W -r "reason" printer-name
```

To disable the printer and cancel the request that is currently printing, type a command in this form:

```
# disable -c -r "reason" printer-name
```

If you include the **-r** option, the *reason* you specify will appear when users ask the status of the printer. If you don't want to specify a reason, you can omit this option.

You can specify multiple printer names on the **disable** command line.

Allowing Users to Enable and Disable Printers

You can permit all users to issue the **enable** and **disable** commands. Doing this allows anyone spotting a problem with a printer to disable it. However, giving users access to these commands can interfere with the proper operation of the print services.

If you want to give users access to the **enable** and **disable** commands, use a standard operating system feature called the "setuid bit." After assigning ownership of these commands to the user **lp** (this should have been done automatically when you installed the software) and setting the setuid bit, all users will be able to use the **enable** and **disable** commands. Clearing the setuid bit removes this privilege.

To allow everybody to run **enable** and **disable**, enter the following two commands:

```
# chown lp /usr/bin/enable /usr/bin/disable

# chmod u+s /usr/bin/enable /usr/bin/disable
```

The first command makes the user **lp** the owner of the commands. This step might be redundant, but it is safer to run the command than to skip it. The second command turns on the setuid bit.

To prevent other users from running **enable** and **disable**, enter the following command:

```
# chmod u-s /usr/bin/enable /usr/bin/disable
```

5.3.8 Displaying Printer Status and Configuration Information

You can use the **lpstat** command to view the current status of a printer and to check its configuration. If you issue the short form of the command, as shown below, the system lists status information.

```
# lpstat -p printer-name
```

This information tells you if the printer exists and if it is busy, idle, or disabled. You should see a message similar to one of the following lines.

```
printer printer-name now printing request-id. enabled since date.

printer printer-name is idle. enabled since date.

printer printer-name disabled since date.
        reason
```

```
printer printer-name waiting for auto-retry.
        reason
```

The `waiting for auto-retry` message means the LP print service failed in trying to use the printer (because of the *reason* shown) and will try again later.

The long form of the command produces a complete configuration listing as well as status information.

```
# lpstat -p printer-name -l
```

The configuration listing contains the following information.

```
Form mounted: form-name
Content types: content-type-list
Printer type: printer-type
Description: comment
Connection: connection-info
Interface: path-name
On fault: alert-method
After fault: fault-recovery
Users allowed:
     user-list
Forms allowed:
     form-list
Banner required
Character sets:
     character-set-list
Default pitch: integer CPI, integer LPI
Default page size: scaled-decimal-number wide, scaled-decimal-number long
Default port settings: stty-option-list
```

5.4 Troubleshooting

If you are having difficulty getting your printer to work, here are a few suggestions that might help you correct the problem.

5.4.1 No Output – Nothing Prints

The printer is idle; nothing happens. First, check the documentation that came with the printer to see if there is a self-test feature you can invoke. Make sure the printer is working before continuing.

Is the Printer Connected to the Computer?

Check the connections between the computer and the printer.

Is the Printer Set Up Properly?

Set up the printer as described in the "Printer Management" section of this chapter. If you receive error messages when doing this, follow the fixes suggested in the messages.

When you finish setting up the printer, issue the following commands, where *printer-name* is the name you assigned to the printer.

```
# /usr/lib/accept printer-name

# enable printer-name
```

Now submit a sample file for printing:

```
$ lp -d printer-name -T printer-type filename
```

If you haven't specified a printer type for the printer, omit the **-T** *printer-type* option.

Is the Baud Rate Correct?

If the baud rate (the rate at which data is transmitted) is not the same for both the computer and the printer, the printer might not be able to print.

5.4.2 Illegible Output

The output from the printer is unreadable.

Is the Baud Rate Correct?

Usually, when the baud rate of the computer doesn't match that of the printer, you will receive some output, but it will not look like what you submitted for printing. Random characters will appear, with an unusual mixture of special characters and unlikely spacing.

Read the printer documentation to determine the appropriate baud rate. If the printer is connected through a parallel port, the baud rate doesn't matter.

By default, the LP print service uses 9600 baud. If the printer uses a different baud rate, you can set that baud rate for the LP print service. Enter a command in the following form:

/usr/lib/lpadmin –p *printer-name* **–o stty=***baud-rate*

Now submit a sample file for printing.

Is the Parity Setting Correct?

Some printers use a parity bit to ensure that the data received for printing has not been garbled in transmission. The parity bit can be encoded in several ways; the computer and the printer must use the same method. If they do not agree, some characters either will not be printed or will be replaced by other characters. Generally, though, the output will look approximately correct, with the spacing of words typical for your document and many letters in their correct places.

Check the printer documentation to see the parity setting the printer expects. By default, the LP print service does not set the parity bit. If the printer sets the parity bit, you can use one of the following commands to set the bit for the LP print service.

The first command sets odd parity generation, the second command sets even parity. The last command sets no parity, which is the default value.

/usr/lib/lpadmin –p *printer-name* **–o stty=oddp**

/usr/lib/lpadmin –p *printer-name* **–o stty=evenp**

/usr/lib/lpadmin –p *printer-name* **–o stty=–parity**

If you also need to set the baud rate, you can include it in the same command, as shown in the following example.

```
# /usr/lib/lpadmin -p printer-name -o "stty='evenp 1200'"
```

Both double and single quotes are needed.

Tabs Set Correctly?

If the printer doesn't expect to receive tab characters, the output can contain the complete contents of the file, but the text that was tabbed might appear against the right margin. See "No Left Margin/Runs Together/Text at Right Margin," later in this section, for more information.

Correct Printer Type?

Refer to "Wrong Character Set or Font," later in this section.

5.4.3 Legible Printing, but Wrong Spacing

The output contains all of the expected text and is readable, but the text is in the wrong format: double-spaced, no left margin, run together, or zig-zagging down the page. These problems can be fixed by adjusting the printer settings (if possible) or by having the LP print service use settings that match those of the printer.

Double-Spaced

Either the printer's tab settings are wrong or the printer is adding a linefeed after each carriage return. (The LP print service adds a carriage return to each linefeed, so the combination causes two linefeeds.)

The **–tabs** option tells the LP print service not to send tabs.

```
# /usr/lib/lpadmin -p printer-name -o stty=–tabs
```

The **–onlcr** option tells the LP print service not to add a carriage return.

```
# /usr/lib/lpadmin -p printer-name -o stty=–onlcr
```

No Left Margin/Runs Together/Text at Right Margin

The printer's tab settings are incorrect; they should be set every 8 spaces. The **–tabs** option tells the LP print service not to send tabs.

```
# /usr/lib/lpadmin -p printer-name -o stty=–tabs
```

Text Zig-Zags Down the Page

The **onlcr** option is needed. This is set by default, but you might have cleared it accidentally.

```
# /usr/lib/lpadmin -p printer-name -o stty=onlcr
```

A Combination of Problems

If you need to use several of these options to take care of multiple problems, you can combine them in a single command. Include the baud rate or parity settings.

```
# /usr/lib/lpadmin -p printer-name -o "stty='-onlcr -tabs 2400'"
```

Both double and single quotes are needed.

Correct Printer Type?

Refer to the next section, "Wrong Character Set or Font."

5.4.4 Wrong Character Set or Font

If the wrong printer type was selected when you set up the printer, the wrong control characters can be sent to the printer. The results are unpredictable. Output might disappear or be illegible, making it look like a problem described earlier in this section. The wrong control characters could also cause the printer to set the wrong character set or font.

If you don't know which printer type to specify, use one of the following procedures to locate the appropriate type.

- If you think the printer type has a certain name, try the following command, replacing *printer-type* with the name.

  ```
  $ TERM=printer-type tput longname
  ```

  This command prints a short description of the printer identified by the *printer-type*. Continue to try the names you think might be right until you find one that identifies your printer.

- If you don't know what names to try, you can examine the */usr/lib/terminfo* directory to see the available names. Enter the following command to examine the directory.

  ```
  $ ls -R /usr/lib/terminfo/*
  ```

Pick names from the list that match a word or number identifying your printer. Try each of the names in the **TERM=** command shown earlier.

When you locate the correct printer type, enter a command in this form:

```
# /usr/lib/lpadmin -p printer-name -T printer-type
```

5.4.5 Dial-Out Failures

The LP print service uses the Basic Networking Utilities to handle dial out printers. If a dialing failure occurs and you are receiving printer fault alerts, the LP print service reports the same error reported by the Basic Networking software for similar problems. (If you haven't arranged to receive fault alerts, they will be mailed to the user **lp**.)

5.4.6 Idle Printers

If a printer is idle but enabled and print requests are queued for it, one of the following conditions might exist:

- The print requests need to be filtered. Slow filters run one at a time to avoid overloading the system. Until a print request has been filtered (if it needs slow filtering), it will not print. Use the following command to see if the first request in the queue is being filtered.

  ```
  $ lpstat -o -l
  ```

- The printer has a fault. After detecting a fault, the LP print service waits about five minutes, then tries to resume printing. It continues trying until a request prints successfully. You can force an immediate retry by enabling the printer.

  ```
  # enable printer-name
  ```

- A dial-out printer is busy or doesn't answer, or all dial-out ports are busy. The LP print service waits five minutes before trying to reach a dial-out printer again. If the dial out printer can't be reached for an hour or two (depending on the reason), the LP print service finally alerts you to a possible problem. You can force an immediate retry by enabling the printer.

  ```
  # enable printer-name
  ```

- Lost child process. If the process controlling the printer is killed (by the operating system during periods of extremely heavy load, or by an administrator), the LP print service might not realize it for a few minutes. Disabling the printer and then reenabling it will force the LP print service to check for the controlling process and then to restart it. Make sure the printer is really idle before you do this. If a printer is disabled while it is printing a request, the request will have to be reprinted in its entirety.

  # **disable** *printer-name*

  # **enable** *printer-name*

  If the lost process is controlling a slow filter, don't try to reenable the printer. Instead, put the print request at the head of the queue on hold. Then resume it, as shown below. (The first command list the requests in the queue.)

  # **lpstat –o –l**

  # **lp –i** *request-id* **–H hold**

  # **lp –i** *request-id* **–H resume**

5.5 Managing the Printing Load

At times you will need to tell the LP print service to reject print requests for a specific printer. For example, you might need to take the printer out of service for maintenance purposes. When you do this, you should first move any print requests queued for that printer to another printer in your system.

When you are ready to bring the printer back into service, you can tell the LP print service to start accepting requests for the printer.

5.5.1 Moving Requests to Another Printer

The LP print service allows you to move print requests from one printer queue to another. If you want to spread out the printing load among the printers in your system, you can move individual print requests. If a printer will be going out of service, you should move all of its print requests to another printer.

To move specific print requests to another printer, type a command in this form:

```
# /usr/lib/lpmove request-id printer-name
```

You can include more than one *request-id* on the command line.

To move all print requests from one printer to another, type a command in the following form. *printer-name1* is the printer where the requests are queued, *printer-name2* is the printer to which the requests will be moved.

```
# /usr/lib/lpmove printer-name1 printer-name2
```

When you use this form of the command, the LP print service stops accepting requests for the first printer. (The **reject** command also tells the LP print service to stop accepting requests.)

5.5.2 Rejecting Requests for a Printer or Class

The **reject** command tells the LP print service to stop accepting print requests for a specific printer or class. The command has this form:

```
# /usr/lib/reject -r "reason" printer-or-class-name
```

If you include more than one printer or class on the command line, use spaces to separate the names. The **-r** option allows you to specify a reason for rejecting print requests. The reason appears whenever anyone tries to print a file on the printer. If you don't want to specify a reason, omit this option.

Although the **reject** command prevents new print requests from being accepted, it will not move or cancel any requests currently queued for the printer. Use **lpmove** to move the requests.

5.5.3 Accepting Requests for a Printer or Class

When you want a printer or class to start accepting print requests, enter a command in this form:

```
# /usr/lib/accept printer-or-class-name
```

You can specify more than one printer or class on the command line.

This command tells the LP print service to start accepting requests for the printer or class you specified. You should issue this command in the following situations:

- A printer that has been rejecting requests is ready to return to service.
- You have added a new printer or class to the LP service.

5.5.4 Examples

The following examples use the **lpmove**, **reject**, and **accept** commands.

Example 1

You need to perform some preventive maintenance on printer *lp1*. To prevent the loss of print requests, first move all requests for printer *lp1* to printer *lp2*. After moving the requests, the LP print service will no longer accept requests for *lp1* (you would obtain the same result by running the **reject lp1** command before the **lpmove** command).

```
# /usr/lib/lpmove lp1 lp2
```

At this point, you can disable the printer.

Example 2

lp1 is now ready to return to service. The following command tells the LP print service to begin accepting print requests for *lp1*.

```
# /usr/lib/accept lp1
```

If you disabled the printer earlier, you can now reenable it.

Example 3

You notice that someone has queued several large files for the printer *laser1*. Meanwhile, *laser2* is idle. Move the two largest requests (`laser1-23` and `laser1-46`) to *laser2* and reject any new requests for *laser1* for the time being.

```
# /usr/lib/lpmove laser1-23 laser1-46 laser2

# /usr/lib/reject -r "too busy--will reopen later" laser1
```

5.6 Managing Printer Queues

The LP print service allows users to assign a priority level to each print request. In the print queue, requests having a higher priority are placed ahead of requests having a lower priority. This section describes commands that allow you to control the priority levels for print requests submitted to a print queue. You can limit the priority levels that individual users can specify for a print request and you can set a default priority level. You can also move a print request within a queue or put it on hold.

5.6.1 Printer Queue Setup

The print service uses priority levels to adjust the position of a print request in the queue. Users can assign a priority level to each print request they submit. A priority level is a number from 0 to 39. 0 is the highest priority; 39 is the lowest priority. In the print queue, requests having a higher priority go ahead of requests having a lower priority.

You can assign a priority limit to any of the users on your system. Users cannot specify a priority level higher than their limit. However, they can specify a lower priority. You can also set up a default priority limit that applies to all users who were not assigned a personal limit. For example, you might want to allow some users to specify priority level 1 for their requests; you would assign a priority limit of 1 to those users. You could then set a default priority limit for the rest of the users on your system, preventing them from requesting a priority level above that limit.

You can also set a default priority level. This level applies when users do not specify priority levels in their print requests.

Setting User Priority Limits

The **lpuser** command sets a priority level for specific users. If */usr/lib* is not in your path, you must type the entire pathname (**/usr/lib/lpuser**) when you use this command.

To set a priority limit for one or more users, type a command in the following form. (*priority-limit* is a number from 0 to 39; 0 is the highest priority.) If you specify more than one user name, use commas or spaces to separate the names. (If you use spaces, enclose the list of names in quotes.)

```
# lpusers -q priority-limit -u users
```

If you want to set a priority limit for all other users, enter a command in the following form, where *priority-level* is a number from 0 to 39.

```
# lpusers -q priority-limit
```

This limit applies to all users who have not been assigned a personal priority limit.

If you want to change a user's priority limit, reenter the **lpusers** command and specify a new limit. If you want to remove a priority limit for one or more users, enter a command in this form:

```
# /usr/lib/lpusers -u users
```

The default priority limit now applies to these users.

Setting a Default Priority Level

If a user does not specify a priority level for a print request, the print service applies the default priority level to the request. The initial default priority level is 20. However, you can change this priority level if you want. To do this, type a command in the following form, where *priority-level* contains the level you want to use as the default.

```
# lpusers -d priority-level
```

Don't confuse this priority level with the default priority limit. The default priority level applies when a user doesn't specify a priority level. The default priority limit applies if you haven't assigned a priority limit to a user.

Examining Priority Limits

The following command allows you to examine the settings you have assigned with the **lpusers** command.

```
# /usr/lib/lpusers -l
```

5.6.2 Moving a Request in the Queue

You can move a print request within a print queue in the following ways:

- Adjust the priority of a print request to any level, regardless of the user's priority limit.
- Put a print request on hold.
- Move a print request to the head of the queue.

The following paragraphs describe each of these procedures.

Changing the Priority for a Request

If you want to move a print request up or down in the print queue, you can assign a new priority level to it. The LP print service will then move the print request so that it will be ahead of lower-priority requests and behind requests having the same or a higher level. This step overrides the priority limit assigned to the user as well as the default priority limit.

To change the priority level for a print request, type a command in this form:

```
# lp -i request-id -q priority-level
```

You can change only one request at a time.

Putting a Request on Hold

You can put any print request on hold. If the request is currently printing, the printing will stop. Users can place only their own requests on hold.

To place a request on hold, enter a command in this form:

```
# lp -i request-id -H hold
```

When you are ready to print the request, use the **resume** command to reinstate the request. Users can reinstate requests they have put on hold, but they cannot reinstate requests the system administrator put on hold. To reinstate a request, enter a command in the following form:

```
# lp -i request-id -H resume
```

The request will now move up the queue and will eventually print. If the request was printing when you put it on hold, it will be the next request to print.

Normally, printing begins on page one; you can, however, start the printing on a later page. Enter a command in the following form if you want to start printing on another page:

```
# lp -i request-id -H resume -P page-number-
```

The final dash indicates that all pages following the page you specified are to print.

To print only part of a file, your site must be using a filter having this capability. The default filter used by the LP print service does not have this capability. If your site is not using an appropriate filter, the print service will reject attempts to resume printing on a later page.

Moving a Request to the Head of the Queue

You can also move a print request to the head of the queue. If you want the print request to start printing immediately but another request is currently printing, place the first request on hold, as described in the previous section.

To move a print request to the head of the queue, type the **lp** command and include the **–H** option followed by the word **immediate**.

```
# lp –i request-id –H immediate
```

Only the system administrator can move a print request to the head of the queue. If you use the **–H immediate** option for more than one print request, the last request to be moved to the head of the queue will print first.

5.7 Forms Setup

The print service can keep track of print requests requiring a preprinted form (such as checks, invoices, letterhead, or labels) and can alert you to mount the form.

This section tells how to define a form to the print service, how to specify the notification method used when a form needs to be mounted, how to restrict user access to a form, and how to delete a form from the print service.

"Mounting a Form or Print Wheel," earlier in this chapter, describes how to tell the print service that forms have been loaded and it can print the appropriate print requests.

5.7.1 Defining a Form

When you add a new form, you must use the **lpforms** command to define its characteristics. The print service uses this information to initialize the printer and to send you reminders about the form.

You will need to provide the following information:

Page length
: The length of the form, or of each page in a multipage form. You can specify the length as a number of lines or as the size in inches or centimeters.

Page width
: The width of the form, expressed in columns, inches, or centimeters.

Number of pages
: The number of pages in a multipage form.

  If a filter is available, the print service uses this number to restrict the alignment pattern to the length of one form. If no filter is available, the LP print service does not truncate the output from the alignment pattern.

Line pitch
: A measurement specifying the number of lines either per inch or per centimeter.

Character pitch
: A measurement specifying the number of characters either per inch or per centimeter.

Character set choice
: The character set, print wheel, or font cartridge used to print this form. You can allow the user to specify a different character set (or print wheel or font cartridge) or you can require that a specific character set (or print wheel or font cartridge) be used.

Ribbon color
: If the form requires a specific color of ribbon, the LP print service can remind you to use that ribbon when you mount the form.

Comment
: Any comment you want to make about the form.

Alignment pattern
: This is a sample of the text that will print on the form. It allows you to determine if the forms are lined up properly in the printer. If necessary, you can adjust the forms in the printer and then repeat the alignment pattern.

  The print service does not try to mask sensitive information in an alignment pattern. If you do not want sensitive information to print on sample forms, you should mask the appropriate data. Only the users **lp** and **root** can read the file containing the alignment pattern.

You can record this information in a file and then use the file as input to the **lpforms** command, or you can type each piece of information on the command line. With either method, you must enter the information in the following format:

```
Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice: character-set-name mandatory
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern:
alignment-pattern
```

With two exceptions, the information can appear in any order. The exceptions are the alignment pattern (which must always appear last) and the *comment* (which must always follow the line with the `Comment:` prompt). If the *comment* contains a line beginning with a key phrase (such as `Page`

`length`, `Page width`, and so on), begin the line with a greater-than character (>). This character does not appear when you view the comment on the screen.

You don't need to specify all of the information. If you don't specify an item, the print service assigns the default value, as shown below.

`Page length`	66 lines
`Page width`	80 columns
`Number of pages`	1
`Line pitch`	6
`Character pitch`	10
`Character set choice`	any
`Ribbon color`	any
`Comment`	(no default)
`Alignment pattern`	(no default)

If you write a form definition in a file, use a command such as the following to supply the definition to the **lpforms** command. The *form* name can be up to 14 characters long and can include letters, digits, and underscores.

```
# /usr/lib/lpforms -f form -F file
```

If you want to supply the form definition on the command line, type a command in the following form. The *form* name can be up to 14 characters long and can include letters, digits, and underscores.

```
# /usr/lib/lpforms -f form -
```

After typing the final hyphen, go to the next line and begin typing the form definition.

If you need to change a form, just reenter the appropriate command. You need to provide information only for the items that will change.

5.7.2 Viewing a Form Definition

After you set up a form, you can use the **lpforms** command to look at its definition.

```
# lpforms -f form -l
```

You can also use the **lpstat** command to look at a form definition. When you type the command, you will receive a status message regarding the form.

```
# lpstat -f form
```

If you add the **-l** option, the definition of the form also appears. (The display does not include the alignment pattern.)

```
# lpstat -f form -l
```

5.7.3 Removing a Form Definition

If you are no longer using a form, you should delete the definition for that form. To do this, use the **-x** option to the **lpforms** command.

```
# lpforms -f form -x
```

5.7.4 Restricting Access to a Form

If it is necessary to restrict access to a form, you can specify the users allowed to print that specific form. If you don't set up this option, all users can issue print requests for the form.

You can use an allow list and a deny list to specify the users able to print the form. The allow list contains names of users allowed to print the form; the deny list contains users that cannot print it. If there are entries in the allow list, the print service ignores the deny list. If there are no entries in the allow list, the print service uses the deny list. If both lists are empty, everyone can use the form.

To add user names to the allow list, type a command as shown below. If a user name was already in the deny list, it will be removed from that list. To allow all users to request the form, specify **allow:all**.

```
# lpforms -f form -u allow:users
```

To add user names to the deny list, type a command as shown below. If a user name was already in the allow list, it will be removed from that list. If you don't want regular users to be able to request the form, type **deny:all**. In this case, only **lp** and **root** will be able to request the form.

lpforms -f *form* -**u deny:***users*

If you specify multiple user names, use commas or spaces to separate the names. If you use spaces, enclose the list (including **allow:** or **deny:**) in quotes.

5.7.5 Notification to Mount a Form

When a user creates a print request for a particular form, the request is held in the print queue until you tell the print service that you have mounted the form.

The print service can notify you when a print request asks for a particular form. The **-A** option to the **lpforms** command lets you specify how you want to be alerted: by a mail message, by a message on your screen, or by another command. You can also choose not to receive alerts. If you do not specify an alert method, you will not receive an alert when a form needs to be mounted. Instead, you must monitor the queue for this type of print request.

In the following commands, the **-Q** option indicates the number of print requests that must be waiting for the form before you are alerted. The **-W** option indicates the number of minutes between alerts. If you want to receive just one alert, omit this option.

- To receive a mail message (via the **mail** program), type a command in this form:

  # **lpforms** -f *form* -**A mail** -**Q** *integer* -**W** *minutes*

- To have a message written to your terminal (via the **write** command), type a command in this form:

  # **lpforms** -f *form* -**A write** -**Q** *integer* -**W** *minutes*

- To specify that another command be run, type a command in this form:

  # **lpforms** -f *form* -**A** *'command'* -**Q** *integer* -**W** *minutes*

  The shell environment in effect when you enter this command is saved and then restored when *command* executes. If you want another person to be notified when a form must be mounted, type the appropriate **-A** option:

  -A 'mail *user'*

  -A 'write *user'*

- If you don't want to receive alerts, type a command in this form:

  # **lpforms** –f *form* **–A none**

If you are receiving repeated alerts, you can issue the following command to tell the print service to stop sending alerts. If you are receiving alerts for more than one form, you can stop all of the alerts by typing **all** as the *form*.

lpforms –f *form* **–A quiet**

Chapter 6
TTY Management

6.1 Introduction 6-1
Terms 6-1

6.2 The TTY System 6-2
How the TTY System Works 6-2
Displaying Line Settings 6-3
Creating New Line Settings and Hunt Sequences 6-4
Modifying TTY Line Characteristics 6-4
Setting Terminal Options 6-6

Chapter 6
TTY Management

6.1 Introduction

This chapter describes the TTY subsystem. Chapter P6 in the first half of the guide contains the procedures you can use to maintain the system.

This chapter defines TTY terms and describes how the TTY system works. It also tells how to perform these tasks:

- View existing line settings.
- Create new line settings and hunt sequences.
- Modify TTY line characteristics.
- Set terminal options.

6.1.1 Terms

The following terms are used in this chapter:

TTY. Derived from the abbreviation for teletypewriter, this term covers the whole area of access between the operating system and peripheral devices, including the system console. It shows up in commands such as **getty**(1M) and **stty**(1), in the names of device special files such as */dev/ttyGA/GAAA*, and in the names of files such as */etc/gettydefs*, which is used by **getty**.

TTY line. The physical equipment through which access to the computer is made.

port. A synonym for TTY line.

line settings. A set of line characteristics.

bit rate. The speed at which data is transmitted over the line.

mode. The characteristics of the terminal interface. The TTY line and the terminal must be working in the same mode before communication can take place. Described in **termio**(7).

hunt sequence. A circular series of line settings. During the login sequence, a user looking for a compatible connection to the computer can go from one setting to the next by sending a BREAK signal.

terminal options. Selectable settings that define the way a given terminal operates. Described in **termio**(7).

6.2 The TTY System

The remaining sections in this chapter describe how the TTY system operates and tell how you can administer it.

6.2.1 How the TTY System Works

A series of four processes (**init**(1M), **getty**(1M), **login**(1), **sh**(1)) connects a user to the operating system. **init** is a general process spawner that is invoked as the last step in the boot procedure. Guided by the */etc/inittab* file, **init** spawns a **getty** process for each line on which users can log in.

The **getty** command requires a **line** argument. This argument is the name of a special file in the */dev* directory. For a description of other arguments that can be used with **getty**, see the *Reference Manual.*

A user attempting to make a connection generates a request-to-send signal that is routed by the hardware to the **getty** process for one of the TTY line files in */dev*. (We're omitting how the signal gets from the user's terminal to the computer.) **getty** responds by sending an entry from the */etc/gettydefs* file down the line. The *gettydefs* entry used depends on the **speed** argument provided with the **getty** command. (In the SYNOPSIS of the **getty**(1M) command the argument name is **speed**, but it is really a pointer to the **label** field of a *gettydefs* entry.) If no **speed** argument is provided, **getty** uses the first entry in *gettydefs*.

Among the fields in the *gettydefs* entry (described later in this chapter) is the login prompt. On receiving the login prompt, the user enters a login name. **getty** starts **login**, using the login name as an argument. **login** issues the prompt for a password, evaluates the user's response, and, assuming that the password is acceptable, calls in the shell program listed in the */etc/passwd* entry for the user. If no shell is named, **/bin/sh** is furnished by default. **login** also executes */etc/profile*.

/bin/sh executes the user's *.profile* (or *.login*), if it exists. This file often contains **stty** commands to reset terminal options. The connection between the user and the operating system has now been made.

6.2.2 Displaying Line Settings

There are two ways to check existing line settings:

1. Use the system administration menus, specifically the **sysadm**(1) **lineset** subcommand. This command first shows the full range of line settings, then lets you examine a line in detail.
2. Look directly in */etc/gettydefs*.

The */etc/gettydefs* file contains information used by the **getty**(1M) command to establish the speed and terminal settings for a line. The general format of the *gettydefs* file is as follows:

```
label# initial-flags # final-flags #login-prompt #next-label
```

Following are a few lines from a sample *gettydefs* file.

```
4800H# B4800 # B4800 SANE IXANY TAB3 HUPCL #login: #9600H

9600H# B9600 # B9600 SANE IXANY TAB3 HUPCL #login: #19200H

19200H# B19200 # B19200 SANE IXANY TAB3 HUPCL #login: #2400H

2400H# B2400 # B2400 SANE IXANY TAB3 HUPCL #login: #1200H

1200H# B1200 # B1200 SANE IXANY TAB3 HUPCL #login: #300H

300H# B300 # B300 SANE IXANY TAB3 HUPCL #login: #4800H
```

These entries form a single, circular hunt sequence; the last field on each line is the label of the next line. The next-label field for the last line points back to the first line in the sequence. The object of the hunt sequence is to link a range of line speeds. If you see garbage characters instead of a clear login prompt, entering a BREAK causes **getty** to step to the next entry in the sequence. The hunt continues until the communication rate of the line matches the speed of the user's terminal. The flag fields shown have the following meanings:

B300-B19200	The communication rate (bits-per-second) of the line.
HUPCL	Hang up on close.
SANE	A composite flag that stands for a set of normal line characteristics.
IXANY	Allow any character to restart output. If this flag is not specified, only DC1 (Ctrl-Q) will restart output.
TAB3	Send tabs to the terminal as spaces.

For a description of all **getty** flags, see **termio**(7).

6.2.3 Creating New Line Settings and Hunt Sequences

There are two ways to create line setting and hunt sequences:

1. Use the system administration menus, specifically the **sysadm mklineset**(1) subcommand. This command leads you through a series of prompts; your responses make up the information for a new *gettydefs* entry.
2. Use **ed**(1) or **vi**(1) to edit */etc/gettydefs*.

If you want to use an editor to create new lines for the *gettydefs* file, use the existing lines as examples. Each entry in the file is followed by a blank line. After editing the file, run this command:

```
# /etc/getty -c /etc/gettydefs
```

This command causes **getty** to scan the file and then print the results on your terminal. Any unrecognized modes or improperly constructed entries are reported.

6.2.4 Modifying TTY Line Characteristics

There are two ways to modify TTY line characteristics:

1. Use the system administration menus, specifically the **sysadm modtty**(1) subcommand. This command leads you through a series of prompts; your responses edit a "getty" entry in */etc/inittab*.
2. Use **ed**(1) or **vi**(1) to edit */etc/inittab*.

The */etc/inittab* file contains instructions for the **/etc/init**(1M) command. The general format of a line entry in the */etc/inittab* file is as follows.

identification:level:action:process

Each line contains four fields separated by colons:

identification	An identifier (1 to 4 characters long) for the line entry.
level	The run level in which the entry is to be performed.
action	How **/etc/init** treats the process field (refer to the **inittab**(4) manual page for complete information).
process	The shell command to be executed.

/etc/inittab contains several entries that spawn **getty** processes. Following are some sample entries:

```
co:234:respawn:/etc/getty syscon console sqnt200
GAAA:2:respawn:/etc/getty ttyAA/GAAA 9600 sqnt200
GAAB:2:respawn:/etc/getty ttyAA/GAAB 9600 sqnt200
GAAC:2:respawn:/etc/getty ttyAA/GAAC 9600 sqnt200
GAAD:2:off:/etc/getty ttyAA/GAAD 9600;#line not in use
GAAE:2:off:/etc/getty ttyAA/GAAE 9600;#line not in use
```

Following are some changes you might want to make to an *inittab* entry for a TTY line:

- Change the action. Two actions that apply to TTY lines are "respawn" and "off" (see the **inittab**(4) manual page for complete information on this field).
- Add or change arguments in the process field. For example, you can add the terminal type, as shown in the preceding examples. If you do this, the system can determine the terminal type when a user logs in on the line.

  Another frequently used argument is **–t***nn*. This tells **getty** to hang up if nothing is received within *nn* seconds. It's good practice to use the **–t** argument on dial-up lines.
- Add or change comments. Comments can be inserted after a command. Type a semicolon (;) to end the command, then type a pound sign (#) followed by the comment.

6.2.5 Setting Terminal Options

The TTY system establishes a basic style of communication between the user's terminal and the operating system. After logging in, the user can specify additional terminal options.

The **stty** command controls terminal options. Many users add an **stty** command to their *.profile* (or *.login*) so the options they want are automatically set as part of the **login** process. Here is an example of a simple **stty** command.

```
$ stty cr0 nl0 echoe -tabs erase ^H
```

The options in the example have these meanings:

cr0 nl0 No delay for carriage return or newline. Delays are not used on a video display terminal, but are necessary on some printing terminals to allow time for the mechanical parts of the equipment to move.

echoe Erases characters as you backspace.

-tabs Expand tabs to spaces when printing.

erase ^H Change the delete character to Ctrl-H. The default delete character is the pound sign (#). Because most terminals transmit Ctrl-H when BackSpace is pressed, specifying this option makes BackSpace useful.

Chapter 7
Basic Networking

7.1 Introduction 7-1

7.2 Networking Hardware 7-1

7.3 Networking Commands 7-2
User Programs 7-2
Administrative Programs 7-3

7.4 Daemons 7-4

7.5 Supporting Database 7-5
Devices File 7-6
Protocols 7-11
Dialers File 7-11
Systems File 7-14
Dialcodes File 7-18
Permissions File 7-18
How Entries are Structured 7-18
Permissions File Options 7-19
MACHINE Entry for Other Systems 7-24
Combining MACHINE and LOGNAME Entries 7-25
Poll File 7-25
Devconfig File 7-26
Sysfiles File 7-26
Other Networking Files 7-27

7.6 Administrative Files 7-28

7.7 Direct Links 7-30
Creating ***Devices*** File Entries 7-31
Making Changes to the ***/etc/inittab*** File 7-31
Making ***Systems*** File Entries 7-32

Chapter 7
Basic Networking

7.1 Introduction

This chapter describes the Basic Networking system. Chapter P7 in the first half of this manual contains the procedures you can use to set up and maintain the system.

The basic networking utilities enable computers using the UNIX operating system to communicate with each other and with remote terminals. These utilities range from those used to copy files between computers (**uucp** and **uuto**) to those used for remote login and command execution (**cu**, **ct**, and **uux**).

As an administrator, you need to be familiar with the administrative tools, logs, and database files used by these utilities. Chapter P7, "Basic Networking," tells how to maintain these utilities. This chapter goes into greater detail about the files, directories, daemons, and commands involved with these utilities.

7.2 Networking Hardware

Before your computer can communicate with other computers, you must set up the hardware to complete the communications link. The cables and other hardware you need depend on how you want to connect the computers: over direct links, telephone lines, or local area networks.

Direct Links

You can create a direct link to another computer by running cables between serial ports on the two computers. Direct links are useful when two computers communicate regularly and are physically close—within 50 feet of each other. You can use a limited distance modem to increase this distance somewhat. Transfer rates of up to 19200 bits per second (bps) are possible when computers are directly linked.

Telephone Lines

An Automatic Call Unit (ACU) allows your computer to communicate with other computers over standard telephone lines. The ACU dials the telephone number requested by the networking utilities. The computer it is trying to contact must have a telephone modem capable of answering incoming calls.

Local Area Network

A Local Area Network (LAN) can be the communication medium for basic networking. When your computer is established as a node on a LAN, it can contact any other computer connected to the LAN.

7.3 Networking Commands

Basic networking programs can be divided into two categories: user programs and administrative programs. The following paragraphs describe the programs in each category.

7.3.1 User Programs

The user programs for basic networking are in */usr/bin*. No special permission is needed to use these programs, which are described in the *Reference Manual*.

cu Connects your computer to a remote computer, allowing you to be logged in on both computers at the same time. You can transfer files or execute commands on either computer without dropping the initial link.

ct Connects your computer to a remote terminal, enabling the user of the remote terminal to log in. This command also enables the user on the remote terminal to call the computer and request that the computer call it back. In this case, the computer drops the initial link so that the remote terminal's modem will be available when it is called back.

uucp Lets a user copy a file from one computer to another. This command creates work files and data files, queues the job for transfer, and calls the **uucico** daemon, which in turn attempts to contact the remote computer.

uuto Copies files from one computer to a public spool directory (*/usr/spool/uucppublic/receive)* on another computer. Unlike **uucp**, which lets you copy a file to any accessible directory on the remote computer, **uuto** places the file in an appropriate spool directory and tells the remote user to pick up the file with **uupick**.

uupick Retrieves files that **uuto** has placed in the */usr/spool/uucppublic/receive* directory.

uux Creates the work, data, and execute files needed to execute commands on a remote computer. The work file contains the same information as work files created by **uucp** and **uuto**. The execute files contain the command string to be executed on the remote computer as well as a list of the data files. The data files are those files required for the command execution.

uustat Lists the status of requested transfers (**uucp**, **uuto**, or **uux**). It also provides you with a means of controlling queued transfers.

7.3.2 Administrative Programs

Most of the administrative programs, database files, and shell scripts are in the */usr/lib/uucp* directory. The only exception is **uulog**, which is in */usr/bin*. These commands are described in the *Reference Manual.*

You should use the **uucp** login when you administer the basic networking utilities because it owns the basic networking and spooled data files. The home directory of the **uucp** login is */usr/lib/uucp*. (The other basic networking login, **nuucp**, is used by remote computers to access your computer. Calls from **nuucp** are answered by **uucico**.)

uulog Lists the contents of the log files for a specified computer. Log files are created for each remote system that communicates with your computer. The log files contain records of each use of **uucp**, **uuto**, and **uux**.

uucleanup Cleans up the spool directory. This command is normally executed from a shell script called **uudemon.cleanup**, which is started by **cron**.

Uutry Tests call processing capabilities and does a moderate amount of debugging. This command invokes the **uucico** daemon to establish a communication link between your computer and the remote computer you specify.

uucheck Checks for the presence of basic networking directories, programs, and support files. It can also check certain parts of the *Permissions* file for obvious syntactic errors.

7.4 Daemons

The basic networking utilities include three daemons. (A *daemon* is a routine that runs as a background process and performs a system-wide public function.) These daemons handle file transfers and command executions. They can also be run manually from the shell.

uucico Selects the device used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers data and execute files, logs the results, and notifies the user by **mail** when the transfer completes. When the local **uucico** daemon calls a remote computer, it "talks" to the **uucico** daemon on the remote computer.

After all required files have been created, the **uucico** daemon is executed by the **uucp**, **uuto**, and **uux** programs to contact the remote computer. It is also executed by the **uusched** and **Uutry** programs.

uuxqt Executes remote execution requests. It searches the spool directory for execute files (always named X.*file*) that have been sent from a remote computer. When an X.*file* file is found, **uuxqt** opens it to obtain the list of data files that are required for the execution. It then determines whether the required data files are available and accessible. If the files are present and can be accessed, **uuxqt** checks the *Permissions* file to verify that it has permission to execute the requested command. The **uuxqt** daemon is executed by the **uudemon.hour** shell script, which is started by **cron**.

uusched Schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** determines the order in which remote computers will be called. **uusched** is executed by a shell script called **uudemon.hour**, which is started by **cron**.

UUCP also uses the following internal program.

uugetty This program is very similar to the **getty** program. However, it permits a line (port) to be used in both directions. If you specify "bidirectional" when you use the **sysadm portmgmt** command to modify a port in the */etc/inittab* file, a **uugetty** will be assigned to the port. **uugetty** is executed as a function of the **init** program and is described in the *Reference Manual.*

7.5 Supporting Database

The support files are in the */usr/lib/uucp* directory. You can make most changes to these files by using the system administration menu commands described in Chapter P7, "Basic Network Procedures." The descriptions in this section provide details on the structure of these files so that you can edit them manually.

Devices Contains information about the location and line speed of the automatic call unit, direct links, and network devices.

Dialers Contains character strings required to negotiate with network devices (automatic calling devices) in the establishment of connections to remote computers (non-801-type dialers).

Systems Contains information needed by the **uucico** daemon and the **cu** program to establish a link to a remote computer. This information includes the name of the remote computer, the name of the connecting device associated with the remote computer, when the computer can be reached, its telephone number, and the login and password your system must use to log in on the remote computer.

Dialcodes	Contains dial-code abbreviations that can be used in the phone number field of *Systems* file entries.
Permissions	Defines the level of access granted to computers when they attempt to transfer files or to execute commands on your computer.
Poll	Defines computers that are to be polled by your system and specifies when they are to be polled.
Devconfig	Configures the utilities for a transport provider conforming to the AT&T Transport Interface.
Sysfiles	Assigns different or multiple files to be used by **uucico** and **cu** as *Systems*, *Devices*, and *Dialers* files.

There are several other files that may be considered part of the supporting database, but they are not directly related to the process of establishing a link and transferring files. These files—*Maxuuxqts*, *Maxuuscheds*, and *remote.unknown*—are described briefly in Chapter P7, "Basic Network Procedures."

7.5.1 *Devices* File

The *Devices* file (*/usr/lib/uucp/Devices*) contains information about the devices that can be used to establish a link to a remote computer. These devices include automatic call units, direct links, and network connections.

NOTE

*This file works closely with the **Dialers**, **Systems**, and **Dialcodes** files. Before modifying any of these files, you should be familiar with them all. A change to an entry in one file can require a change to a related entry in another file.*

Each entry in the *Devices* file has this format:

Type Line Line2 Class Dialer-Token-Pairs

The fields in an entry contain this information:

Type This field can contain either a keyword (**Direct** or **ACU**), the name of a Local Area Network switch, or a system name.

- **Direct**: This keyword indicates either a direct link to another computer or a switch (for **cu** connections only).
- **ACU**: This keyword indicates that the link to a remote computer is made through an automatic call unit (Automatic Dial Modem). This modem can be connected either directly to your computer or indirectly through a Local Area Network (LAN) switch.
- *LAN_Switch*: This value can be replaced by the name of a LAN switch. **micom** and **develcon** are the only switches having caller scripts in the *Dialers* file. (You can add your own LAN switch entries to the *Dialers* file.) If you are setting up an AT&T Transport Interface-compatible network, you would use the special dialer types **TLI** or **TLIS**.
- *Sys-Name*: This value indicates a direct link to a particular computer. (*Sys-Name* is replaced by the name of the computer.) This naming scheme specifies that the line associated with this *Devices* entry is for a particular computer in the *Systems* file.

The keyword used in the *Type* field is matched against the third field of *Systems* file entries as shown below:

Devices: **ACU** `ttyGA/GAAH - 1200 penril`

Systems: `eagle Any` **ACU** `1200 3251 ogin: nuucp \`
`ssword: Oakgrass`

You can also designate a protocol to be used for a device. See the "Protocols" section at the end of the description of this file.

Line This field contains the device name of the line (port) associated with the *Devices* entry. For example, if the Automatic Dial Modem for a particular entry were attached to the */dev/ttyGA/GAAH* line, the name entered in this field would be **ttyGA/GAAH**.

Line2 If the keyword **ACU** is used in the *Type* field and the ACU is an 801-type dialer, *Line2* will contain the device name of the 801 dialer. (801-type ACUs do not contain a modem. Therefore, a separate modem is required and would be connected to a different line that is defined in the *Line* field.) This means that one line would be allocated to the modem and another to the dialer. Since non-801 dialers do

not normally use this configuration, they will ignore the *Line2* field, but it must still contain a hyphen (–) as a placeholder.

Class If the keyword **ACU** or **Direct** is used in the *Type* field, *Class* can be just the speed of the device. However, it can contain a letter and a speed (for example, C1200, D1200) to differentiate between classes of dialers (Centrex or Dimension PBX). This is necessary because many larger offices have more than one type of telephone network: one network might be dedicated to serving only internal office communications while another handles the external communications. In such a case, it becomes necessary to distinguish between the lines used for internal communications and the lines used for external communications. The keyword used in the *Class* field of the *Devices* file is matched against the fourth field of *Systems* file entries as shown below:

Devices:
```
ACU ttyGA/GAAH - D1200 penril
```

Systems:
```
eagle Any ACU D1200 3251 ogin: nuucp \
          ssword: Oakgrass
```

If a device can be used at any speed, the keyword **Any** can be used in the *Class* field. If **Any** is used, the line will match any speed requested in a *Systems* file entry. If this field is **Any** and the *Systems* file *Class* field is **Any**, the speed defaults to 1200 bps.

Dialer-Token-Pairs:

This field contains pairs of dialers and tokens. The *dialer* portion can be the name of an automatic dial modem or a LAN switch, or it can be **direct** for a direct link device. You can have any number of Dialer-Token-Pairs. The *token* portion can immediately follow the *dialer* portion. If it is omitted, it will be taken from a related entry in the *Systems* file.

This field has the following format:

dialer token dialer token

The last pair may or may not be present, depending on the associated device (dialer). In most cases, the last pair contains only a *dialer* portion and the *token* portion is retrieved from the *Phone* field of the *Systems* file entry.

A valid entry in the *dialer* portion can be defined in the *Dialers* file or it can be one of several special dialer types. These special dialer

types, which are listed below, are compiled into the software and are available without having entries in the *Dialers* file.

- 801 – Bell 801 autodialer
- TLI – Transport Level Interface Network (without STREAMS)
- TLIS – Transport Level Interface Network (with STREAMS)

The *Dialer-Token-Pairs* (*DTP*) field can be structured in four different ways, depending on the device associated with the entry:

- If an automatic dialing modem is connected directly to a port on your computer, the *DTP* field of the associated *Devices* file entry will have only one pair. This pair would normally be the name of the modem. This name is used to match the *Devices* file entry with an entry in the *Dialers* file. Therefore, the *dialer* field must match the first field of a *Dialers* file entry:

  Devices: `ACU ttyGA/GAAH - 1200 ventel`

  Dialers: `ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!`

  Notice that only the *dialer* portion (`ventel`) is present in the *DTP* field of the *Devices* file entry. This means that the *token* to be passed on to the dialer (in this case, the phone number) is taken from the *Phone* field of a *Systems* file entry (\T is implied). The backslash sequences are described later in this section.

- If a direct link is established to a particular computer, the *DTP* field of the associated entry would contain the keyword **direct**. This is true for both types of direct link entries, **Direct** and *System-Name* (refer to discussion on the *Type* field).

- If a computer you want to reach is on the same local network switch as your computer, your computer must first access the switch. The switch can then make the connection to the other computer. In this type of entry, there is only one pair. The *dialer* portion is used to match a *Dialers* file entry:

  Devices: `develcon ttyGA/GAAI - 1200 develcon \D`

  Dialers: `develcon "" "" \pr\ps\c est:\007 \E\D\e \007`

As shown, the *token* portion is blank, which indicates that it is retrieved from the *Systems* file. The *Systems* file entry for this particular computer will contain the token in the *Phone* field, which is normally reserved for the phone number of the computer (refer to the discussion of the *Phone* field in the *Systems* file). This type of *DTP* contains an escape character (\D), which ensures that the contents of the *Phone* field will not be interpreted as a valid entry in the *Dialcodes* file.

- If an automatic dialing modem is connected to a switch, your computer must first access the switch, which will make the connection to the automatic dialing modem. This type of entry requires two *dialer-token-pairs*. The *dialer* portion of each pair (the fifth and seventh fields of the entry) is used to match entries in the *Dialers* file:

  Devices: ACU ttyGA/GAAJ - 1200 **develcon** vent **ventel**

  Dialers: **develcon** "" "" \pr\ps\c est:\007 \E\D\e \007
  Dialers: **ventel** =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!

  In the first pair, **develcon** is the dialer and **vent** is the token that is passed to the Develcon switch to tell it which device (ventel modem) to connect to your computer. Because each LAN switch can be set up differently, this token would be unique for each switch. After the ventel modem is connected, the second pair is accessed, where **ventel** is the dialer and the token is retrieved from the *Systems* file.

There are two escape characters that can appear in a *DTP* field:

\T Indicates that the *Phone* (*token*) field should be translated using the *Dialcodes* file. This escape character is normally placed in the *Dialers* file for each caller script associated with an automatic dial modem (penril, ventel, and so on). Therefore, the translation will not take place until the caller script is accessed.

\D Indicates that the *Dialcodes* file should not be used to translate the *Phone* (*token*) field. If no escape character is specified at the end of a *Devices* entry, the \D is assumed (default). A \D is also used in the *Dialers* file for entries associated with network switches (develcon and micom).

Protocols

You can define the protocol to use with each device. In most cases, this is unnecessary because you can use the default or define the protocol with the particular system you are calling (see *Systems* file, *type* field). If you do specify the protocol, you must use the form *Type,Protocol*. The available protocols are as follows:

g This protocol is slower and more reliable than **e**. It is good for transmission over noisy telephone lines.

e This protocol is faster than **g**, but it assumes error-free transmission.

For reliable local area networks, you should use the **e** protocol.

7.5.2 *Dialers* File

The *Dialers* file (*/usr/lib/uucp/Dialers*) specifies the initial conversation that must take place on a line before it can be made available for transferring data. This conversation is usually a sequence of ASCII strings that is transmitted and expected. ASCII dialers often use this type of sequence to dial a phone number.

As shown in the earlier examples, the fifth field in a *Devices* file entry is either an index into the *Dialers* file or a special dialer type (801, TLI, or TLIS). Here an attempt is made to match the fifth field in the *Devices* file with the first field of each *Dialers* file entry. In addition, each odd numbered *Devices* field (starting with the seventh position) is used as an index into the *Dialers* file. If the match succeeds, the *Dialers* entry is interpreted to perform the dialer negotiations.

Each entry in the *Dialers* file has the following format:

dialer substitutions expect-send ...

The *dialer* field matches the fifth and any additional odd numbered fields in the *Devices* file. The *substitutions* field is a translate string; for each pair of characters, the first character is mapped to the second character. This is typically used to translate = and – into whatever the dialer requires for "wait for dialtone" and "pause."

The remaining *expect-send* fields are character strings. Below are some character strings distributed in the *Dialers* file.

```
penril =W-P "" \d > s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP > 9\c OK
ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
hayes  =,-, "" \dAT\r\c OK\r \EATDT\T\r\c CONNECT
rixon  =&-% "" \d\r\r\c $ s9\c )-W\r\ds9\c-) s\c : \T\r\c $ 9\c LINE
vadiac =K-K "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE
develcon "" "" \pr\ps\c est:\007 \E\D\e \007
micom ""     "" \s\c NAME? \D\r\c GO
direct
att2212c        =+-,     "" \r\c :--: atol2=y,T\T\r\c red
att4000 =,-,     "" \033\r\r\c DEM: \033s0401\c \006 \033s0901\c \
         \006 \033s1001\c \006 \033s1102\c \006 \033dT\T\r\c \006
att2224 =+-,     "" \r\c :--: T\T\r\c red
nls      ""      "" NLPS:000:001:1\N\c
```

Following are the meanings of some of the escape characters used in the *Dialers* file:

\p	Pause (approximately ¼ to ½ second)
\d	Delay (approximately 2 seconds)
\D	Phone number or token without *Dialcodes* translation
\T	Phone number or token with *Dialcodes* translation
\K	Insert a BREAK
\E	Enable echo checking (for slow devices)
\e	Disable echo checking
\r	Carriage return
\c	No newline or carriage return
\n	Send newline
\nnn	Send octal number

Additional escape characters are listed under "*Systems* File," later in this chapter.

The `penril` entry in the *Dialers* file is executed as follows. First, the phone number argument is translated, replacing any = with a **W** (wait for dialtone) and replacing any – with a **P** (pause). The handshake given by the remainder of the line works as follows:

`""`	Wait for nothing. (In other words, proceed to the next item.)
`\d`	Delay for 2 seconds.
`>`	Wait for a **>**.
`s\p9\c`	Send an **s**, pause for ½ second, send a **9**, send no terminating newline
`)-W\p\r\ds\p9\c-)`	Wait for a **)**. If it is not received, process the string between the – characters as follows. Send a **W**, pause, send a carriage-return, delay, send an **s**, pause, send a **9** without a newline, and then wait for the **)**.
`y\c`	Send a **y**.
`:`	Wait for a **:**.
`\E\TP`	Enable echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.) Then send the phone number. The **\T** means take the phone number passed as an argument and apply the *Dialcodes* translation and the modem function translation specified in the second field of this entry. Then send a **P**.
`>`	Wait for a **>**.
`9\c`	Send a **9** without a newline.
`OK`	Wait for the string **OK**.

7.5.3 *Systems* File

The *Systems* file (*/usr/lib/uucp/Systems*) contains the information needed by the **uucico** daemon to establish a communication link to a remote computer. Each entry in the file represents a computer that can be called by your computer. In addition, the basic networking software can be configured to prevent any computer that does not appear in this file from logging in on your computer (refer to "Other Networking Files," later in this chapter, for a description of the *remote.unknown* file). More than one entry can be present for a particular computer. The additional entries represent alternative communication paths that will be tried in sequential order. The management of this file is supported by the system administration menu subcommand **systemmgmt**.

Using the *Sysfiles* file, you can define several files to be used as "Systems" files. See the description of the *Sysfiles* file for details.

Each entry in the *Systems* file has the following format:

System-Name Time Type Class Phone Login

The fields contain this information:

System-name
This field contains the node name of the remote computer.

Time This field is a string that indicates the day-of-week and time-of-day when the remote computer can be called. The *Time* field has this format:

daytime[*;retry*]

The day portion can be a list containing some of the following items:

Su Mo Tu We Th Fr Sa
For individual days

Wk For any week-day (Mo Tu We Th Fr)

Any For any day

Never For a passive arrangement with the remote computer. If the *Time* field is **Never**, your computer will never initiate a call to the remote computer. The call must be initiated by the remote computer. In other words, your computer is in a passive mode in respect to the remote computer.

The *time* portion should be a range of times such as 0800-1230. If no *time* portion is specified, any time of day is assumed to be allowed for the call. A time range that spans 0000 is permitted. For example, **0800-0600** means that calls can be made at any time except the period between 6 AM and 8 AM. Here is an example:

```
Wk 1700-0800, Sa,Su
```

This example allows calls from 5:00 PM to 8:00 AM, Monday through Thursday, and any time Saturday and Sunday.

An optional subfield, *retry*, is available to specify the minimum time (in minutes) before a retry following a failed attempt. The default wait is 60 minutes. The subfield separator is a semicolon (;). For example, **Any;9** is interpreted as call any time, but if a failure occurs, wait at least 9 minutes before retrying.

Type This field contains the device type that should be used to establish the communication link to the remote computer. The keyword in this field is matched against the first field of *Devices* file entries:

Systems: `eagle Any ACU,g D1200 3251 ogin: nuucp \`
`ssword: Oakgrass`

Devices: `ACU ttyGA/GAAH - D1200 penril`

You can define the protocol used to contact the system by adding it to the *Type* field. The example above shows how to attach the protocol **g** to the device type **ACU**.

Class This field indicates the transfer speed of the device used in establishing the communication link. It can contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers. If the device can be used at any speed, the keyword **Any** can be used. This field must match the *Class* field in the associated *Devices* file entry:

Systems: `eagle Any ACU D1200 NY3251 ogin: nuucp \`
`ssword: Oakgrass`

Devices: `ACU ttyGA/GAAH - D1200 penril`

If information is not required for this field, use a hyphen (–) as a place holder for the field.

Phone This field provides the phone number (token) of the remote computer for use by automatic dialers. The phone number is made up of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be listed in the *Dialcodes* file.

Systems: `eagle Any ACU D1200 NY3251 ogin: nuucp \ ssword: Oakgrass`

Dialcodes: `NY 9=1212555`

In this string, an equals sign (=) tells the ACU to wait for a secondary dial tone before dialing the remaining digits. A dash (–) instructs the ACU to pause 4 seconds before dialing the next digit.

If your computer is connected to a LAN switch, you can access other computers that are connected to that switch. The *Systems* file entries for these computers will not have a phone number in the *Phone* field. Instead, this field will contain the token that must be passed on to the switch so it will know which computer your system wants to contact. (This is usually just the system name.) The associated *Devices* file entry should have a **\D** at the end of the entry to ensure that this field is not translated using the *Dialcodes* file.

Login This field contains login information given as a series of fields and subfields. It has this format:

expect send

expect is the string your system receives. *send* is the string your system sends after receiving the *expect* string.

The *expect* field is made up of subfields in this form:

expect[*-send-expect*]...

The *send* is sent if the prior *expect* is not read successfully and the *expect* following the *send* is the next expected string. For example, **login--login** tells UUCP to expect **login**. If UUCP receives **login,** it will go on to the next field. If it does not receive **login**, it will send nothing followed by a new line, then look for **login** again. If no characters are initially expected from the remote computer, the characters "" (null string) should be used in the first *expect* field. Note that all *send* fields will be sent followed by a newline unless the *send* string is terminated with a **\c**.

Here is an example of a *Systems* file entry that uses an expect-send string:

```
owl Any ACU 1200 Chicago6013 "" \r ogin:-BREAK-ogin: \
uucpx word: xyzzy
```

This example says send a carriage return and wait for `ogin:` (for `Login:`). If you don't get `ogin:`, send a BREAK. When you do receive `ogin:`, send the login name **uucpx**. Then, when you receive `word:` (for `password:`), send the password **xyzzy**.

There are several escape characters that cause specific actions when they are a part of a string sent during the login sequence. The following escape characters are useful in UUCP communications:

`\N` Send or expect a null character (ASCII NUL).

`\b` Send or expect a backspace character.

`\c` If at the end of a string, suppress the newline that is normally sent. Ignored otherwise.

`\d` Delay 2 seconds before sending or reading more characters.

`\p` Pause for approximately ¼ to ½ second.

`\E` Start echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.)

`\e` Echo check off.

`\n` Send a newline character.

`\r` Send or expect a carriage-return.

`\s` Send or expect a space character.

`\t` Send or expect a tab character.

`\\` Send or expect a \ character.

`EOT` Send or expect EOT newline twice.

`BREAK` Send or expect a break character.

`\K` Same as BREAK.

`\ddd` Collapse the octal digits (ddd) into a single character.

7.5.4 *Dialcodes* File

The *Dialcodes* file (*/usr/lib/uucp/Dialcodes*) contains the dialcode abbreviations that can be used in the *Phone* field of the *Systems* file. Each entry has this format:

abb dial-seq

abb is the abbreviation used in the *Phone* field of the *Systems* file; *dial-seq* is the dial sequence passed to the dialer when that particular *Systems* file entry is accessed.

The following entry would work with a *Phone* field such as `jt7867` in the *Systems* file.

```
jt 9=847-
```

If the token in the dialer-token-pair is `\T` when the entry containing `jt7867` is encountered, the sequence 9=847-7867 is sent to the dialer.

7.5.5 *Permissions* File

The *Permissions* file (*/usr/lib/uucp/Permissions*) specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the remote computer's ability to request files and to receive files queued by the local site. Another option specifies the commands that a remote system can execute on the local computer.

How Entries are Structured

Each entry is a logical line; physical lines are terminated by a \ to indicate continuation. Entries are made up of options delimited by white space. Each option is a name/value pair in the following format:

name=value

Note that no white space is allowed within an option assignment.

Comment lines begin with a pound sign (#) and they occupy the entire line up to a newline character. Blank lines are ignored (even within multiline entries).

There are two types of *Permissions* file entries:

LOGNAME
: Specifies the permissions that take effect when a remote computer logs in on (calls) your computer.

MACHINE
: Specifies permissions that take effect when your computer logs in on (calls) a remote computer.

LOGNAME entries contain a LOGNAME option; MACHINE entries contain a MACHINE option.

The following items should be considered when using the *Permissions* file to restrict the level of access granted to remote computers:

- All login IDs used by remote computers to log into your system must appear in one and only one LOGNAME entry.
- If a site is called and its name does not appear in a MACHINE entry, it will have the following default permissions/restrictions:
  a. Local send and receive requests will be executed.
  b. The remote computer can send files to your computer's */usr/spool/uucppublic* directory.
  c. The commands sent by the remote computer for execution on your computer must be one of the default commands (usually **rmail**).

Permissions File Options

This section describes the options available for LOGNAME and MACHINE entries, specifies how they are used, and lists their default values.

REQUEST. When a remote computer calls your computer and asks to receive a file, the request can be granted or denied. The REQUEST option specifies whether the remote computer can set up file transfers from your computer. The following string specifies that the remote computer can ask to transfer files from your computer.

```
REQUEST=yes
```

The following string specifies that the remote computer cannot ask to transfer files from your computer.

```
REQUEST=no
```

This is the default value. It is used if the REQUEST option is not specified. The REQUEST option can appear in either a LOGNAME (remote machine calls you) entry or a MACHINE (you call remote machine) entry. A note on security: If a remote machine calls you and you don't have a unique login and password for that machine, you can't be sure that the machine is who it says it is.

SENDFILES. When a remote computer calls your computer and completes its work, it might attempt to take work your computer has queued for it. The SENDFILES option specifies whether your computer can send the work queued for the remote computer.

The following string specifies that your computer can send work queued for the remote computer as long as it logged in as one of the names specified in the LOGNAME option.

```
SENDFILES=yes
```

This string is mandatory if your computer is in a "passive mode" with respect to the remote computer.

The next string specifies that files queued in your computer will be sent only when your computer calls the remote computer.

```
SENDFILES=call
```

The is the default value for the SENDFILE option. This option is significant only in LOGNAME entries, as MACHINE entries apply when calls are made to remote computers. If the option is used with a MACHINE entry, it will be ignored.

READ and WRITE. These options specify the parts of the filesystem that **uucico** can read from or write to. The READ option is for requesting files, the WRITE option for depositing files. These options can be used with either MACHINE or LOGNAME entries.

The value for these options is a list of pathnames separated by colons. One of the values must be the prefix of any full pathname of a file coming in or going out.

The *uucppublic* directory is the default for both the READ and the WRITE options.

```
READ=/usr/spool/uucppublic
WRITE=/usr/spool/uucppublic
```

The following strings specify permission to access any file that can be

accessed by a local user with "other" permissions.

```
READ=/ WRITE=/
```

To grant permission to deposit files in */usr/news* as well as in the public directory, the following values would be used with the WRITE option:

```
WRITE=/usr/spool/uucppublic:/usr/news
```

If the READ and WRITE options are used, all pathnames must be specified because the pathnames are not added to the default list. For example, if only the */usr/news* pathname were specified in a WRITE option, permission to deposit files in the public directory would be denied.

You should be careful when making directories accessible for reading and writing by remote systems. For example, you probably wouldn't want remote computers to be able to write over your */etc/passwd* file, so */etc* shouldn't be open to writes.

NOREAD and NOWRITE. The NOREAD and NOWRITE options specify exceptions to the READ and WRITE options or defaults. The following strings permit reading any file except those in the */etc* directory (and its subdirectories—remember, these are prefixes) and writing only to the default */usr/spool/uucppublic* directory.

```
READ=/ NOREAD=/etc WRITE=/usr/spool/uucppublic
```

NOWRITE works in the same manner as the NOREAD option. NOREAD and NOWRITE can be used in both LOGNAME and MACHINE entries.

CALLBACK. The CALLBACK option is used in LOGNAME entries to specify that no transaction will take place until the calling system is called back. There are some good reasons to use CALLBACK. From a security standpoint, if you call back a machine you can be sure it is the machine it says it is. Also, if you are doing long data transmissions, you can choose the machine that will be billed for the longer call.

The following string specifies that your computer must call the remote computer back before any file transfers will take place.

```
CALLBACK=yes
```

The default value for the COMMAND option is as follows:

```
CALLBACK=no
```

The CALLBACK option is rarely used. Note that if two sites set this option for each other, a conversation will never get started.

COMMANDS. The COMMANDS option can be hazardous to the security of your system. Use it with extreme care.

The COMMANDS option can be used in MACHINE entries to specify the commands that a remote computer can execute on your computer. Note that COMMANDS is not used in LOGNAME entries; COMMANDS in MACHINE entries define command permissions whether we call the remote system or it calls us.

The following string indicates the default command that a remote computer can execute on your computer.

```
COMMANDS=rmail
```

If a command string is used in a MACHINE entry, the default commands are overridden. For example, the following entry overrides the COMMAND default so that the computers *owl*, *raven*, *hawk*, and *dove* can now execute **rmail**, **rnews**, and **lp** on your computer.

```
MACHINE=owl:raven:hawk:dove \
COMMANDS=rmail:rnews:lp
```

You can also specify the full pathnames of commands:

```
COMMANDS=rmail:/usr/lbin/rnews:/usr/local/lp
```

In this example, the command **rmail** uses the default path. (The default paths for your computer are */bin*, */usr/bin*, and */usr/lbin*.) When the remote computer specifies **rnews** or **/usr/lbin/rnews**, **/usr/lbin/rnews** will be executed regardless of the default path. Likewise, **/usr/local/lp** is the **lp** command that will be executed.

Including ALL in the list means that any command from the remote computer(s) specified in the entry will be executed. If you use this value, you give the remote computer full access to your computer. Be careful. This allows far more access than normal users have.

The following string illustrates two points.

```
COMMANDS=/usr/lbin/rnews:ALL:/usr/local/lp
```

The ALL value can appear anywhere in the string, and the pathnames specified for **rnews** and **lp** will be used (instead of the default) if the requested command does not contain the full pathnames for **rnews** or **lp**.

The VALIDATE option should be used with the COMMANDS option whenever potentially dangerous commands like **cat** and **uucp** are specified with the COMMANDS option. Any command that reads or writes files is potentially dangerous to local security when executed by the UUCP remote execution daemon (**uuxqt**).

VALIDATE. The VALIDATE option is used in conjunction with the COMMANDS option when specifying commands that are potentially dangerous to your computer's security. It provides a degree of verification of the caller's identity. The use of the VALIDATE option requires that privileged computers have a unique login/password for UUCP transactions. An important aspect of this validation is that the login/password associated with this entry be protected. If an outsider gets that information, that particular VALIDATE option can no longer be considered secure. (VALIDATE is merely an added level of security on top of the COMMANDS option. However, it is a more secure way to open command access than ALL.)

Careful consideration should be given to providing a remote computer with a privileged login and password for UUCP transactions. Giving a remote computer a special login and password with file access and remote execution capability is like giving anyone on that computer a normal login and password on your computer. Therefore, if you cannot trust someone on the remote computer, do not provide that computer with a privileged login and password.

The following LOGNAME entry specifies that if a remote computer claiming to be *eagle*, *owl*, or *hawk* logs in on your computer, it must have used the login **uucpfriend**.

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

As can be seen, if an outsider obtains the **uucpfriend** login/password, masquerading is trivial.

But what does this have to do with the COMMANDS option, which only appears in MACHINE entries? It links the MACHINE entry (and COMMANDS option) with a LOGNAME entry for a privileged login. This link is needed because the execution daemon is not running while the remote computer is logged in. In fact, it is an asynchronous process with no knowledge of what computer sent the execution request. Therefore, the real question is: How does your computer know where the execution files came from?

Each remote computer has its own "spool" directory on your computer. These spool directories have write permission given only to the UUCP programs. The execution files from the remote computer are put in its spool directory after being transferred to your computer. When the **uuxqt** daemon runs, it

can use the spool directory name to find the MACHINE entry in the *Permissions* file and to obtain the COMMANDS list, or if the computer name does not appear in the *Permissions* file, the default list will be used.

The following example shows the relationship between the MACHINE and LOGNAME entries:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
COMMANDS=rmail:/usr/lbin/rnews \
READ=/  WRITE=/

LOGNAME=uucpz VALIDATE=eagle:owl:hawk \
REQUEST=yes SENDFILES=yes \
READ=/  WRITE=/
```

The value in the COMMANDS option means that remote mail and **/usr/lbin/rnews** can be executed by remote users.

In the first entry, you must make the assumption that when you want to call one of the computers listed, you are really calling either *eagle*, *owl*, or *hawk*. Therefore, any files put into one of the *eagle*, *owl*, or *hawk* spool directories is put there by one of those computers. If a remote computer logs in and says that it is one of these three computers, its execution files will also be put in the privileged spool directory. You therefore have to validate that the computer has the privileged login **uucpz**.

MACHINE Entry for Other Systems

You might want to specify different options for the computers that are not mentioned in specific MACHINE entries. This can occur when there are many computers calling in and the command set changes from time to time. In the MACHINE entry, the word "OTHER" is used for the computer name.

```
MACHINE=OTHER \
COMMANDS=rmail:rnews:/usr/lbin/Photo:/usr/lbin/xp
```

All options available for the MACHINE entry can also be set for computers that are not mentioned in specific MACHINE entries.

Combining MACHINE and LOGNAME Entries

When the common options are the same, you can combine MACHINE and LOGNAME entries into a single entry. For example, the following entries share the same REQUEST, READ, and WRITE options.

```
MACHINE=eagle:owl:hawk REQUEST=yes \
  READ=/  WRITE=/

LOGNAME=uucpz REQUEST=yes SENDFILES=yes \
  READ=/  WRITE=/
```

These two entries can be merged as follows:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
LOGNAME=uucpz SENDFILES=yes \
  READ=/  WRITE=/
```

7.5.6 *Poll* File

The *Poll* file (*/usr/lib/uucp/Poll*) contains information for polling remote computers. Each entry in the *Poll* file contains the name of a remote computer to call, followed by a Tab (a space won't work), and finally the hours the computer should be called. Entries in the *Poll* file have this form:

sys-name hour ...

For example, the following entry specifies that computer *eagle* should be polled every four hours.

```
eagle  0 4 8 12 16 20
```

The **uudemon.poll** script does not actually perform the poll. It merely sets up a polling work file (always named C.*file*) in the spool directory. This file will be seen by the scheduler, which is started by **uudemon.hour**.

7.5.7 *Devconfig* File

The */usr/lib/uucp/Devconfig* file is used when your computer communicates over a Streams-based transport provider that conforms to the AT&T Transport Interface (TI).

Devconfig entries define the STREAMS modules that are used for a particular TI device. Entries in the *Devconfig* file have this format:

service=*x* **device=***y* **push=***z*[:*z* ...]

x can be **cu**, **uucico**, or both values separated by a colon. *y* is the name of a TI network and must match an entry in the *Devices* file. *z* is replaced by the names of STREAMS modules in the order that they are to be pushed onto the stream. Different modules and devices can be defined for **cu** and **uucp** services.

Following are some sample *Devconfig* entries:

```
service=cu       device=STARLAN  push=ntty:tirdwr:ld0
service=uucico   device=STARLAN  push=ntty:tirdwr:ld0
```

This example pushes **ntty**, then **tirdwr**, then **ld0**. The *Devconfig* file cannot be modified with the system administration menus command **sysadm**. If you want to change the contents of this file, you must use a text editor.

7.5.8 *Sysfiles* File

The */usr/lib/uucp/Sysfiles* file lets you assign different files to be used by **uucp** and **cu** as *Systems*, *Devices*, and *Dialers* files. Here are some cases in which this optional file might be useful.

- You might want different *Systems* files so that requests for login services can be made to different addresses than requests for **uucp** services.
- You might want different *Dialers* files to use different handshaking for **cu** and **uucp**.
- You might want to have multiple *Systems*, *Dialers*, and *Devices* files. The *Systems* file in particular may become large, making it more convenient to split it into several smaller files.

The *Sysfiles* file has this format:

service=*w* **systems=***x*:*x* **dialers=***y*:*y* **devices=***z*:*z*

w is replaced by **uucico, cu**, or by both values separated by a colon. *x* is one or more files that are to be used as the *Systems* file; the filenames must be separated by colons and are read in the order presented. *y* is one or more files to be used as the *Dialers* file. *z* is one or more files to be used as the *Devices* file. Unless a full path is given, each file is assumed to be relative to the */usr/lib/uucp* directory. A backslash-carriage return (\ Return) can be used to continue an entry onto the next line.

Here's an example of using a local *Systems* file in addition to the usual *Systems* file:

```
service=uucico:cu   systems=Systems:Local_Systems
```

If this line is in */usr/lib/uucp/Sysfiles*, then both **uucico** and **cu** will first look in */usr/lib/uucp/Systems*. If the system they are trying to call doesn't have an entry in that file or the entries in the file fail, they will look in */usr/lib/uucp/Local_Systems*.

When different *Systems* files are defined for **uucico** and **cu** services, your machine will store two different lists of systems. You can print the **uucico** list by issuing the **uuname** command. To print the **cu** list, use the **uuname –c** command.

7.5.9 Other Networking Files

There are three other files that impact the use of basic networking facilities. In most cases, the default values are fine and no changes are needed. If you want to change them, however, use an editor such as **ed** or **vi**.

Maxuuxqts
: This file defines the maximum number of **uuxqt** programs that can run at one time.

Maxuuscheds
: This file defines the maximum number of **uusched** programs that can run at one time.

remote.unknown
: This file is a shell script that executes when a machine that is not in any of the *Systems* files starts a conversation. It will log the conversation attempt and fail to make a connection. If you change the permissions on this file so that it cannot execute (by issuing the command **chmod 000 remote.unknown**), your system will accept any conversation requests.

7.6 Administrative Files

The basic networking administrative files are described below. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

TM (temporary data file)

These data files are created under the spool directory (that is, */usr/spool/uucp/X*) when a file is received from another computer. The directory *X* has the same name as the remote computer sending the file. The names of the temporary data files have this format:

TM.*pid.ddd*

pid is a process-ID and *ddd* is a sequential three digit number starting at 0.

When the entire file is received, the *TM.pid.ddd* file is moved to the pathname specified in the work file that caused the transmission. If processing terminates abnormally, the *TM.pid.ddd* file might remain in the *X* directory. These files should be automatically removed by **uucleanup**.

LCK (lock file)

Lock files are created in the */usr/spool/locks* directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The names of lock files have this format:

LCK..str

str is either a device or computer name. These files can remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files will be ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock.

C. (work file)

Work files are created in a spool directory when work (file transfers or remote command executions) has been queued for a remote computer. The names of work files have this format:

C.sysnxxxx

sys is the name of the remote computer, *n* is the ASCII character representing the grade (priority) of the work, and *xxxx* is the four-digit

job sequence number assigned by UUCP. Work files contain the following information:

- Full pathname of the file to be sent or requested
- Full pathname of the destination or *~user/file* name
- User login name
- List of options
- Name of associated data file in the spool directory (if the **uucp –c** or **uuto –p** option was specified, a dummy name (*D.0*) is used)
- Mode bits of the source file
- Login name of remote user to be notified upon completion of the transfer

D. (Data file)

Data files are created when the command line specifies that the source file is to be copied to the spool directory. The names of data files have the following format:

D.systmxxxxyyy

systm is the first five characters in the name of the remote computer and *xxxx* is a four-digit job sequence number assigned by **uucp**. The job sequence number can be followed by a subsequence number, *yyy*, that is used when there are several *D.* files created for a work (*C.*) file.

X. (Execute file)

Execute files are created in the spool directory prior to remote command executions. The names of execute files have the following format:

X.sysnxxxx

sys is the name of the remote computer, *n* is the character representing the grade (priority) of the work, and *xxxx* is a four-digit sequence number assigned by UUCP. Execute files contain the following information:

- Requester's login and computer name

- Name of file(s) required for execution
- Input to be used as the standard input to the command string
- Computer and filename to receive standard output from the command execution
- Command string
- Option lines for return status requests

7.7 Direct Links

Direct links are beneficial in these situations:

- It is not possible to link the computers together through a Local Area Network (LAN).
- Two computers transfer large amounts of data on a regular basis.
- Two computers are located no more than several hundred feet apart.

The distance between two directly linked computers is dependent on the environment in which the cable is run. The standard for RS-232 connections is 50 feet or less with transmission rates as high as 19200 bits per second (bps). As the cable length is increased, noise on the lines may become a problem, which means that the transmission rate must be decreased or limited-distance modems must be placed on each end of the line.

Do not use more than 1000 feet of cable to connect the two computers or communications will be unreliable. This link should operate comfortably at 9600 bps in a clean (noise free) environment.

Ideally, systems that have a direct link should be running the same release of the operating system to have the full set of capabilities available. However, lack of a common version of the operating system will not prevent you from using the basic networking utilities.

The following support files must be updated to reflect the presence of a direct link:

- */usr/lib/uucp/Devices*

- */etc/inittab*
- */usr/lib/uucp/Systems*

All necessary additions/modifications to these three files can be accomplished by using the system administration menu subcommand **uucpmgmt.**

Creating *Devices* File Entries

The *Devices* file contains the information concerning the location (line) and transmission rate of the link. Entries can be added to the *Devices* file by using the system administration menu subcommand **uucpmgmt devicemgmt** and then selecting the **add** operation. You will be prompted for the following information:

- Port name (for */dev/ttyGA/GAAH* use **ttyGA/GAAH**)
- Device type to call on (**direct**)
- Speed at which you want to call (**9600** or **19200**).

Making Changes to the */etc/inittab* File

There are two versions of the basic networking utilities. The differences between them are reflected in the */etc/inittab* file. The newest version allows for bidirectional login capability by respawning **uugetty** instead of **getty**. This means that if two computers (both using **uugetty**) were connected via a direct link, either of these computers could request communication with the other. This would not be true if only one computer was capable of respawning **uugetty**.

If the direct link is connecting your computer with a computer that has the new version of basic networking, the */etc/inittab* files on both computers should be set up to allow "bidirectional" traffic on the associated lines. This means that the lines used must respawn **uugetty** on each end of the link. This would allow either computer to call the other computer.

If the direct link is connecting your computer with a computer that does not have the new version of basic networking, the */etc/inittab* file would be set up differently on each system. The */etc/inittab* file on each computer would be set up to allow either "incoming" or "outgoing" traffic on its line. If one computer allows incoming traffic, the other must allow only outgoing traffic. A **uugetty** could not be used on either computer in this case.

A computer's */etc/inittab* entry would be respawning **getty** for incoming traffic, or have "respawn" turned off for outgoing traffic. In order for this type of link to work, one of the computers must be set up to poll the other. If the remote computer is allowing only incoming traffic, you must set up your computer to poll the remote computer. (You can use the system administration menu subcommand **uucpmgmt pollmgmt** to do this). If the remote computer is allowing only outgoing traffic on the link, it must be set up to poll your computer.

Entries in the */etc/inittab* file can be changed by using the system administration menu subcommand **uucpmgmt portmgmt** and then selecting the **modify** operation. The **modify** operation will prompt you for the following information:

- Port name you want to modify (for */dev/ttyGA/GAAH*, use **ttyGA/GAAH**)
- Direction of traffic on port (**bidirectional**, **incoming**, or **outgoing**)
- Transmission speed of the link (**9600** or **19200**)

The modify procedure lists the ports that are currently dedicated for use by UUCP (included in *Devices* file). The port name to be modified must appear on this list.

Making *Systems* File Entries

An entry must be made in the *Systems* file for the computer associated with the direct link. This can be done by using the system administration menu subcommand **systemmgmt** and then selecting the **add** operation. The add operation will prompt you for the following information:

- Node name of system
- Type of device to call on (**direct**)
- Transmission speed of link (**9600** or **19200**)
- Device port used with link (for */dev/ttyGA/GAAH* use **ttyGA/GAAH**)
- Login ID used to login on system (**nuucp** or some other login you set up)
- Password used by above login

To prevent possible problems when logging on at high speeds, you can insert pauses (**\p**) between the characters being sent out for the login ID and the password. For example, instead of **nuucp**, you should enter

n\pu\pu\pc\p\pp when prompted for the login ID. Do not select the default by pressing Return. The same applies for the password assigned.

Upon completion of the **add** operation, a new entry for the remote computer is added to the *Systems* file. You must also make an additional entry in the *Devices* file. When you use **devicemgmt** to create an entry for the link, it creates an entry similar to the one shown below.

```
Direct ttyGA/GAAH - 9600 direct
```

Appendix A
Directories and Files

A.1 Directories A-1
A.2 Files A-1

Appendix A
Directories and Files

This appendix describes directories and files of interest to a system administrator.

A.1 Directories

The root filesystem contains these directories:

bin	Contains public commands.
dev	Contains special files that define all of the devices on the system.
etc	Contains administrative programs and tables.
lib	Contains public directories.
lost+found	Used by **fsck** to save disconnected files.
mnt	Used to mount filesystems temporarily.
save	Used by the menu system for saving data.
tmp	Used for temporary files.
usr	Used to mount the */usr* filesystem.

A.2 Files

The following directories and files are important in the administration of your computer.

/etc/checklist

The */etc/checklist* file is used to define a default list of filesystem devices to be checked for consistency by **/etc/fsck** and **/etc/ncheck**. The character (raw) device partition for the filesystem should be identified. The devices listed normally correspond to those mounted when the system is in multiuser mode (run level 2). The *root* filesystem should not be listed in this file. Remember that, with the exception of *root*, a filesystem must be unmounted to be checked. Therefore, the *checklist* file is a convenience for use when in the

single-user mode of operation with only the *root* filesystem mounted. When the system is delivered, this file is empty. (See **checklist**(4) for additional information.)

/etc/fstab

The */etc/fstab* file is used as an argument to the **/etc/mountall** command. The *fstab* file specifies the filesystems to be mounted by **/etc/mountall**. The format of the file is the block device name followed by the mountpoint name. (See **mountall**(1M) for additional information.)

/etc/gettydefs

The */etc/gettydefs* file contains information that is used by **/etc/getty** to set the speed and terminal settings for a line. The **getty** command uses a label to access the *gettydefs* file.

/etc/group

The */etc/group* file describes each group to the system. An entry is added for each new group. Each line in the file consists of four fields separated by colons:

group name:password:group id:login names

Explanations for these fields are as follows:

group name
: The first field defines the group name, which can be from 3 to 6 characters long. The first character is alphabetic; the remaining characters are alphanumeric. You must use lowercase letters.

password
: The second field contains the encrypted group password. This password contains 13 bytes (characters), but the actual password is limited to a maximum of 8 bytes. The encrypted password can be followed by a comma and up to 4 more bytes of password aging information. The use of group passwords is discouraged.

group id
: The third field contains the group identification number, which must be between 0 and 60000. Group identification numbers 0 through 99 are reserved; 0 indicates the superuser (root). Commas are not entered in this field.

login names
The fourth field contains a list of all login names in the group. The names in the list are separated by commas. These users can issue the **/etc/newgrp** command to become a member of the group.

/etc/init.d Directory

The */etc/init.d* directory contains executable files used in upward and downward transitions to all system run levels. These files are linked to files beginning with *S* (start) or *K* (stop) in */etc/rcn.d*, where *n* is the appropriate run level. Files are not executed from this directory. They are executed only from */etc/rcn.d* directories.

/etc/inittab

The */etc/inittab* file contains instructions for the **/etc/init** command. The instructions define the processes that are to be created or terminated for each initialization state (also called a *run level*).

By convention, run levels 1, S, and s are single-user mode; run level 2 is multiuser mode. Chapter 3, "Processor Operations," summarizes the various run levels and describes their uses. (See **inittab**(4) for additional information.) An */etc/inittab* entry contains a series of fields separated by colons.

identification:run-state:action:process

Explanations for these fields are as follows:

identification	The identification field is an identifier for the line entry. It can be from 1 to 4 characters long.
run-state	The run-state defines the run level in which the entry is to be processed.
action	The action field defines how **/etc/init** treats the process field. (Refer to **inittab**(4) for complete information.)
process	The process field defines the shell command that is to be executed.

/etc/motd

The */etc/motd* file contains the message-of-the-day. This message is output by the */etc/profile* file after a user logs in successfully. This message should be short and to the point. The */usr/news* files should be used for longer messages.

/etc/passwd

The */etc/passwd* file identifies each user to the system. An entry is added for each new user. Each line in the file consists of seven fields separated by colons.

login name:passwd:user:group:account:login directory:program

Explanations for these fields are as follows:

login name
: The first field defines the login name, which can be up to 8 characters long. The first character is alphabetic; the remaining characters are alphanumeric. You cannot use uppercase characters.

passwd
: The second field contains the encrypted password, which consists of 13 bytes (characters). The actual password is limited to a maximum of 8 bytes. The encrypted password can be followed by a comma and up to 4 more bytes of password aging information. If your site is using the */etc/shadow* file, the password information is stored in that file and this field contains an x.

user id
: The third field contains the user identification number, which must be between 0 and 60000. Group identification numbers 0 through 99 are reserved; 0 indicates the superuser (root).

group id
: The fourth field contains the group identification number, which must be between 0 and 60000. Group identification numbers 0 through 99 are reserved; 0 indicates the superuser (root).

account
: The fifth field is for miscellaneous information such as the user's name and phone number.

login directory
The sixth field defines the full pathname of the login directory.

program
The seventh field defines the program to be executed after login. If it is null, the Bourne shell (**/bin/sh**) is invoked.

/etc/profile

The */etc/profile* file contains the default profile for all Bourne and Korn shell users. This profile establishes the standard (default) environment for these users. The system administrator can modify this file to set options for the **root** login.

/etc/rc0

The */etc/rc0* file contains a shell script that is executed by **/etc/shutdown** for transitions to single-user state. It is also executed by **/etc/init** for transitions to run levels 0, 5, and 6. Files in the */etc/shutdown.d* and */etc/rc0.d* directories are executed when **/etc/rc0** is run. The file *K00ANNOUNCE* in */etc/rc0.d* prints the message `System services are now being stopped`. If you want a task to execute when the system is taken to run levels 0, s, 5, or 6, add a file to the */etc/shutdown.d* directory.

/etc/rc0.d Directory

The */etc/rc0.d* directory contains files executed by **/etc/rc0** for transitions to system run levels 0, 5, and 6. Files in this directory are linked from the */etc/init.d* directory and begin with either a *K* or an *S*. *K* indicates processes that are stopped when entering run levels 0, 5, or 6; *S* indicates processes that are started.

/etc/rc2

The */etc/rc2* file contains a shell script that is executed by **/etc/init** on transitions to run level 2 (multiuser state). Executable files in */etc/rc.d* and any executable files beginning with *S* or *K* in the */etc/rc2.d* directories are executed when **/etc/rc2** is run. All files in *rc2.d* are linked from files in the */etc/init.d* directory. The following are descriptions of some of those files. These files are prefixed with an *S* or a *K* and a number in the */etc/rc2.d* directory.

MOUNTFILESYS
Sets up and mounts filesystems. Builds the mount table and mounts the **root** (/) and user (*/usr*) filesystems. Makes the */usr/tmp* directory and cleans up (deletes) any previous files in that directory.

cron
Starts the **cron** daemon by executing */etc/cron*.

sysetup
Removes the */etc/ps_data* file to force the **/bin/ps** command to read the */unix* file. Outputs the system configuration if the **/etc/prtconf** command exists. Outputs the system trademark information.

uucp
When basic networking is added to the system, the *uucp* file is added to this directory. The *uucp* file deletes **uucp** locks (LCK*), status files (STST*), and temporary files (TM*) under the */usr/spool/uucp* directory structure.

lp When line printer spooling is added to the system, the *lp* file is added to the *rc.d* directory. The *lp* file removes the spooler lock file and starts the scheduler.

Other files can also be added to the */etc/rc2.d* and */etc/rc.d* directories as a function of adding hardware or software to the system.

/etc/rc2.d Directory

The */etc/rc2.d* directory contains files executed by **/etc/rc2** for transitions to system run level 2. Files in this directory are linked from the */etc/init.d* directory, and begin with either a *K* or an *S*. *K* indicates processes that should be stopped, and *S* indicates processes that should be started.

/etc/rc.d Directory

The */etc/rc.d* directory contains executable files that perform functions needed to initialize the system to run level 2. The files are executed when **/etc/rc2** is run. This directory is maintained only for compatibility reasons.

/etc/save.d Directory

The */etc/save.d* directory contains files that are used by the System Administration Menu commands associated with backing up data. The following files are included:

except A list of the directories and files that should not be copied as part of a backup (**savefiles**) is maintained in this file.

timestamp/ ...
The date and time of the last backup (full or incremental) is maintained for each filesystem in the */etc/timestamp* directory.

/etc/shadow

The */etc/shadow* file contains encrypted passwords and aging information for each user on the system. Only the user **root** can read this file.

/etc/shutdown

The */etc/shutdown* file contains a shell script to shut down the system gracefully in preparation for system backup or scheduled downtime. After stopping all nonessential processes, the **shutdown** script executes files in the */etc/shutdown.d* directory by calling **/etc/rc0** for transitions to run level s or S. For transitions to other run levels, the **shutdown** script calls **/etc/init**.

/etc/shutdown.d Directory

The executable files in */etc/shutdown.d* perform the various functions required during the transition to the single-user state (run levels 1, s, or S). This directory is maintained only for compatibility reasons.

/etc/TIMEZONE

The */etc/TIMEZONE* file sets the time zone environment variable **TZ**. This variable is set initially through the System Administration **setup** function. The **TZ** variable in the *TIMEZONE* file is changed by the System Administration **timezone** command (**sysadm timezone**). The **TZ** variable can be redefined on a user (login) basis by setting the variable in the associated *.profile*. The *TIMEZONE* file is executed by **/etc/rc2**. The format of **TZ** is as follows:

```
TZ=TTT#SSS
```

Explanations for the fields are as follows:

TTT The three-character abbreviation for the local time zone.

The number of hours that the local time zone differs from Greenwich Mean Time (GMT). This field can be entered as a positive or negative number.

SSS The three-character abbreviation for the local daylight savings time zone. This field is entered only if daylight savings time is observed.

/etc/utmp

The */etc/utmp* file contains information on the run-state of the system. This information is accessed by the **who –a** command.

/etc/wtmp

The */etc/wtmp* file contains a history of system logins. The owner and group for this file must be **adm**; the access permissions must be 664. Because this file is updated whenever **login** is run, it increases in size. Periodically, you should clear or truncate this file. The command line **>/etc/wtmp**, when executed by **root**, creates the file with nothing in it. The following command line limits the size of the */etc/wtmp* file to the last 3600 characters in the file:

```
tail –3600c /etc/wtmp > /tmp/wtmp; mv /tmp/wtmp /etc/wtmp
```

Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the *wtmp* file. To use one of these functions, add the appropriate command line to the */usr/spool/cron/crontab/root*, */etc/shutdown.d/...*, or */etc/rc.d/, rc2.d* file.

/usr/adm/sulog

The */usr/adm/sulog* file contains a history of substitute user (**su**) command usage. As a security measure, this file should not be readable by others. The */usr/adm/sulog* file should be truncated periodically to keep the size of the file within a reasonable limit. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the *sulog* file. To use one of these functions, add the appropriate command line to the */usr/spool/cron/crontab/root*, */etc/shutdown.d/...*, or the */etc/rc.d/, rc2.d* file. The following command lines limit the size of the log file to the last 100 lines in the file:

```
tail –100 /usr/adm/sulog > /tmp/sulog
mv /tmp/sulog /usr/adm/sulog
```

/usr/lib/cron/log

A history of all actions taken by **/etc/cron** is recorded in the */usr/lib/cron/log* file. You should truncate this file periodically to keep it within a reasonable size. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the */usr/lib/cron/log* file. To use one of these functions to limit the size of the file, add the appropriate command line to the */usr/spool/cron/crontab/root*, */etc/shutdown.d/...*, or */etc/rc.d/ rc2.d* file, as applicable. The following command line limits the size of the log file to the last 100 lines in the file:

```
tail -100 /usr/lib/cron/log > /tmp/log; mv /tmp/log /usr/lib/cron/log
```

/usr/lib/help/HELPLOG

A history of all actions taken by the **/usr/bin/help** command is kept in the */usr/lib/help/HELPLOG* file. This file is copied to */usr/lib/help/oHELPLOG* and a new */usr/lib/help/HELPLOG* file is created by the **/usr/lib/help/helpclean** command. Executing the **helpclean** command twice in succession zeros out the *HELPLOG* and the *oHELPLOG* files. Note that **/etc/cron**, **/etc/rc0**, or **/etc/rc2** can be used to clean up the *HELPLOG* file. To use one of these functions, add the appropriate command line to the */usr/spool/cron/crontab/root*, */etc/shutdown.d/...*, or */etc/rc.d/ rc2.d* file, as applicable. The following command lines limit the size of the log file to the last 100 lines in the file:

```
tail -100 /usr/lib/help/HELPLOG > /tmp/help
mv /tmp/help /usr/lib/help/HELPLOG
```

/usr/lib/spell/spellhist

A history of all words that **spell**(1) fails to match is kept in the */usr/lib/spell/spellhist* file. Periodically, this file should be reviewed for words that should be added to the dictionary. After the *spellhist* file is reviewed, it can be cleared.

/usr/news

The */usr/news* directory contains news files. The filenames are descriptive of the contents of the files; they are analogous to headlines. When a user uses the **news** command to read news items, an empty file named *.news_time* is created in his or her login directory. The date (time) of this file is used by the **news** command to determine if a user has read the latest news files.

/usr/spool/cron/crontabs

The */usr/spool/cron/crontabs* directory contains crontab files for the **adm**, **root**, and **sys** logins. Providing their lognames are in the */usr/lib/cron/cron.allow* file, users can establish their own *crontabs* file using the **crontab** command. If the *cron.allow* file does not exist, the */usr/lib/cron/cron.deny* file is checked to determine if the user is denied the use of the **crontab** command.

As **root**, you can use the **crontab**(1) command or edit the appropriate file under */usr/spool/cron/crontabs* to make the desired entries. Revisions to the file take effect at the next reboot. An entry in */usr/spool/cron/crontabs/logname* has this format:

minute hour day month day-of-week command

The fields have these meanings:

minute	The minutes field is a one- or two-digit number in the range 0 through 59.
hour	The hour field is a one- or two-digit number in the range 0 through 24.
day	The day field is the numerical day of the month in the range 1 through 31.
month	The month field is the numerical month of the year in the range 1 through 12.
day-of-week	The day-of-week field is the numerical day of the week where Sunday is 0, Monday is 1, and Saturday is 6.
command	The command field is the program or command that is executed at the time specified by the first five fields.

The following syntax applies to the first five fields:

- Two numbers separated by a hyphen indicates an inclusive range of numbers between the two specified numbers.
- A list of numbers separated by commas specifies all of the numbers listed.
- An asterisk specifies all legal values.

In the command field (sixth field), a percent sign (%) is translated to a newline character. Only the first line of a command field (character string up to the percent sign) is executed by the shell. Any other lines are made available to the command as standard input.

Glossary

Archive. See *Backup*.

Automatic call unit (ACU). A hardware device that can dial telephone numbers; it is usually built into a modem. The UUCP system uses automatic call units to call a remote system over telephone lines.

Backup. To make a copy of a file, directory, or filesystem. If you back up a directory or filesystem, the backup procedure preserves the directory structure.

Bit rate. The speed (in bits-per-second) at which data is transmitted over a TTY line (port).

Block. A unit to measure (in bytes) the data transfer size. A block must be a multiple of the sector size. A filesystem block typically contains either 4096 or 8192 bytes.

Block device. A device that handles data in blocks. When the operating system needs to read or write data from a block device, it transfers blocks of data (in 2 Kbyte units) into or from a buffer. Disk drives are block devices.

Boot. To start up the operating system.

Boot block. The first 8K block on a disk. This block can contain programs for booting the operating system. Each filesystem also contains a boot block.

Character device. A device that handles data one character at a time, as a stream of bytes. Terminals, printers, and modems are character devices.

Class. A set of printers having common attributes. When a user sends a print request to a class, the first available printer in that class prints the request.

Console. A terminal connected directly to the computer. It is used for communications between the operator and the computer.

Cylinder. A set of tracks on a disk. The read/write head can access all tracks within a cylinder without moving.

Cylinder group. Several contiguous cylinders on a disk. The filesystem spreads files and directories throughout its cylinder groups and attempts to store data for a file within a single cylinder group.

Daemon. A routine that runs as a background process and performs a system-wide function. Daemons are typically scheduled by the **cron** program.

Default value. Data that appears in a field when you make a menu selection.

Device. Disk drives, tape drives, cartridge drives, printers, modems, and so on.

Device driver. A special-purpose program used to control a specific set of devices. Device drivers are typically part of the operating system.

Device special files. Files in the */dev* directory that represent devices. Device special files are the operating system interface to the appropriate device drivers.

Dial code. Abbreviations that represent a portion of a telephone number (used by UUCP).

Direct link. Hardware that allows computers located near each other to communicate. Cables are run between the serial ports of the computers, enabling UUCP conversations to occur.

Disk sector. A unit to measure the storage capacity of a disk. A disk sector contains 512 bytes, which is the smallest indivisible transfer size for disks.

Filesystem. A hierarchical collection of files and directories that begins with a root directory. The root filesystem is the source of the operating system.

Filesystem label. A label that identifies a filesystem. It has two parts: an *fsname* that corresponds to the directory mountpoint and a *volume*, which is a secondary identifier.

Firmware mode. A run level in which all filesystems are unmounted, the operating system is not running, and only the firmware monitor is active. This run level is used to run standalone programs such as diagnostics. It is also the only run level where it is safe to remove power from the system.

Firmware monitor. A part of the system that runs tests, initializes devices, and carries out other procedures necessary to bring up the operating system.

Format. Creating an addressing scheme on a disk.

Fragment. A portion of a file block.

Free blocks. Blocks in a filesystem that are not being used by a file.

Full backup. A backup that copies an entire filesystem or the complete contents of specific directories.

getty. A process that waits at a TTY port for a user to log in. It determines the appropriate line setting for the port, including the communication speed and the login prompt, and then starts up the **login** process.

Group. A set of users having access to a common set of files. Groups are defined in the */etc/group* file, which includes the name of the group, the group ID, and the names of the people belonging to the group.

Hunt sequence. A series of line settings that enable the system to try different communication rates when a user logs in. The user presses Break to cycle through the line settings.

Incremental backup. A backup that copies only new files and those files that have been changed since the last full backup.

Indirect blocks. Data blocks that are not referenced directly by an inode. These data blocks contain portions of a large file.

init. A process invoked during the boot procedure. It reads the */etc/inittab* file to determine the processes to be started up at each run level.

Inode. The part of a filesystem that stores information about a particular file or directory. The information includes the owner, group, permissions, file size, links, the time the file was last accessed or modified, and the location of disk blocks storing data in the file.

Interface program. A program that prepares a print request for printing on a specific printer.

Line setting. A set of line characteristics, such as the communication speed, that adjust a TTY line. These characteristics allow the computer to communicate with a specific peripheral device.

LP scheduler. A program that controls the LP print service. It picks up print requests and sends them to the appropriate printer queue.

Master system. A system that initiates a UUCP conversation.

Menu. A box on the screen that lists a group of related submenus or options. These options correspond to operating system procedures.

Mount. To make a filesystem accessible to users.

Multiuser mode. An operating system mode that allows all users access to the system.

Node name. The official name of a machine in a network. This name is usually the same as the *system name*.

Partition. A portion of a disk. A partition can contain a filesystem, swap space, a raw database, or other types of information.

Password aging. A security feature that allows you to specify when a password must be changed.

Port. The physical equipment through which interaction between the computer and peripheral devices takes place.

Prompt. The questions that appear when you select a menu system subcommand. The prompts tell you what the system needs to know to run a subcommand.

Public directory. Any directory that allows read/write permission for all users. Typically, files transferred by UUCP are placed in a public directory (the default is */usr/spool/uucppublic*).

Raw device. A block device that handles data as a character device (as a stream of bytes, rather than a block). Programs such as **fsck** and **volcopy** access disk and tape drives as raw devices.

Reboot. To halt the system, then start it up again. This procedure initializes the state of the operating system.

Respawn. To continuously restart a process after it has exited. This method is used to make TTY lines available for logins after users have logged out.

Restore. The process of recovering files, directories, or filesystems from a backup medium.

Run level. Operating system states (the most commonly-used states are single-user mode and multiuser mode).

Root filesystem. The first filesystem in the operating system. It begins with a directory called *root* (indicated by a /) and contains several system directories as well as the operating system kernel.

Single-user mode. An operating system mode that allows only the superuser to access the system from the console.

Slave system. A system that receives a UUCP call.

Slice. See *Partition*.

Spawn. To create a process. For example, at boot time, the **init** process spawns a **getty** process for each TTY port that can accept logins.

Submenu. A menu that appears as an option on another menu.

Superblock. The part of a filesystem that stores information about the filesystem, including its name, the number of data blocks it contains, and the maximum number of files it can contain. The superblock is created when the filesystem is created.

Superuser. A privileged user who uses the **root** login. This login owns many system files and has access to all parts of the system.

System default destination. A printer or class that receives print requests not directed to a specific printer.

System name. A name for the system. This name is usually the same as the *Node name*.

Track. A portion of a disk platter that contains data.

TTY line. See *Port*.

Unmount. To make a filesystem inaccessible to users.

User account. An entry in the */etc/passwd* file that defines a user's login name, user ID, home directory, and shell program. The user's password, which is stored in the */etc/shadow* file, is also part of a user account.

uugetty. A program similar to **getty** that allows a TTY line to be used for both logins and outgoing calls. This program allows modems to accept incoming calls and to make outgoing calls from the same physical port.

VTOC (volume table of contents). The part of a disk that contains a listing of the partitions on the disk, their sizes, and their starting points.

Index

A

accept command 5-28,39
Accept print requests 5-28; P5-5
adduser, default values P2-4
adm login 1-8
Administrative commands 1-7
Administrative logins P1-5
Alerts, printers 5-16,20,49
Alignment pattern 5-46
Automatic call unit 7-1

B

Backups ..... 4-15 to 17
  complete 4-15; P4-6
  directories P4-10
  files P4-10
  filesystems P4-6
  incremental 4-16; P4-8
  schedule reminders 4-17; P4-14
Bad blocks 4-24
Banner page 5-23
Basic networking
  administrative programs 7-3
  daemons 7-4
  data files 7-29
  debugging P7-18
  Devconfig file P7-11
  Devices file P7-3
  direct links 7-30
  execute files 7-29
  files ..... 7-5; P7-1 to 12
  hardware 7-1
  lock files 7-28
  log files P7-17
  maintenance P7-14

Maxuuscheds file P7-13
Maxuuxqts file P7-13
Permissions file P7-11
Poll file P7-9
remote.unknown file P7-13
Sysfiles file P7-12
Systems file P7-7
temporary data files 7-28
user commands 7-2
UUCP logins P7-12
work files 7-28
bin login 1-8
Bit rate 6-1
Blocks 4-4
Boot block 4-7
Broadcast messages P2-11

C

Character sets 5-14
Check filesystems ..... 4-20 to 25; P4-3
Classes 5-25
accept print requests 5-28,39
delete from print service 5-27
reject print requests 5-38
Complete backups 4-15; P4-6
Configure printers 5-6
Corrupted filesystems 4-21
cpio command 4-18
ct command 7-2
cu command 7-2
Cylinder groups 4-6
Cylinders ..... 4-4 to 6

D

daemon login 1-8
Daemons, UUCP 7-4
Data blocks 4-8
Data files, UUCP 7-29
date command P1-2
Date, set P1-1

Devconfig file 7-6,26; P7-11
Devices file 7-5
add devices P7-3
protocol 7-11
setup 7-6
df command 4-12
Dialcodes file 7-6
setup 7-18
Dialers file 7-5
escape characters 7-12
setup 7-11
Direct links 7-1,30
Directories
backup P4-10
monitoring size 4-13
restore from backup P4-12
Directory 4-1
disable command 5-29
Disable printers 5-29
Disk drives
blocks 4-4
cylinders 4-6
disk sectors 4-4
partitions 4-2
tracks 4-4
Disk sectors 4-4
Disk usage 4-12; P4-5
monitor file and directory size 4-13
inactive files 4-13
percent of disk space used 4-12
user limits 4-14
du command 4-14
Duplicate blocks 4-23

E

enable command 5-29
Enable printers 5-29; P5-5
Environment 2-6
/etc/checklist file A-1
/etc/fstab file A-2
/etc/gettydefs file 6-3; A-2

/*etc*/*group* file 2-5; A-2
add group P2-3
change entry P2-5
delete entry P2-7
list entries P2-9
/*etc*/*init.d* directory A-3
/*etc*/*inittab* file ..... 3-8 to 10; 6-4; P7-5
/*etc*/*motd* file 2-10; A-4
/*etc*/*passwd* file 1-3; 2-2; A-4
add user P2-1
change entry P2-6
delete user 2-4; P2-8
list users P2-10
password aging 1-3
/*etc*/*profile* file 2-7; A-5
/*etc*/*rc.d* directory A-6
/*etc*/*rc0* file A-5
/*etc*/*rc0.d* directory A-5
/*etc*/*rc2* file A-5
/*etc*/*rc2.d* directory A-6
/*etc*/*save.d* directory A-6
/*etc*/*save.d*/*except* file 4-17
/*etc*/*shadow* file 1-3; 2-3; A-7
add user P2-1
delete user 2-4; P2-8
password aging 1-3
/*etc*/*shutdown* file A-7
/*etc*/*TIMEZONE* file A-7
/*etc*/*utmp* file A-8
/*etc*/*wtmp* file A-8
Execute files, UUCP 7-29
export command 2-6

F

Faults, printer ..... 5-20 to 22
Files
backup P4-10
inactive 4-13
monitoring size 4-13
restore from backup P4-12
system 3-3

Filesystems ..... 4-1 to 25
administration policies 4-11
backup 4-15; P4-6
bad blocks 4-24
boot block 4-7
check ..... 4-19,20 to 25; P4-3
connectivity 4-25
corruption 4-21
data blocks 4-8
directories in *root* A-1
disk usage 4-12; P4-5
duplicate blocks 4-23
fragments 4-9
incremental backup 4-16; P4-8
indirect blocks 4-9
inodes 4-8
lost+found directory 4-25
mount P4-2
preen 4-20
restore from backup 4-18; P4-12
root 3-3
shell scripts for administration 4-11
superblock 4-7
unmount P4-2
Filters ..... P5-18 to 22
add P5-20
change P5-21
delete P5-22
find command 4-13
Firmware mode 3-7,14
Font cartridges 5-14
Forms, print service ..... P5-11 to 17
add ..... 5-45 to 47; P5-13
alerts 5-49
change P5-15
delete 5-48; P5-17
mount 5-49
print 5-19
restrict access 5-48
Fragments 4-9
Free blocks, check 4-22
fsck command ..... 4-20 to 25; P4-3

bad blocks 4-24
check free blocks 4-22
check inodes ..... 4-22 to 25
check superblock 4-22
duplicate blocks 4-23
interactive mode 4-21
lost+found directory 4-25
non-interactive mode 4-21
fsstat command 4-20

G

Groups 2-5
add P2-3
change information P2-5
delete P2-7
ID 2-2; P2-4

H

Home directory 2-3
default value P2-4
Hunt sequence 6-2,3; P6-2,4

I-J-K

Incremental backups 4-16; P4-8
Indirect blocks 4-9
init process ..... 3-8 to 10; P3-4
Inodes 4-8
check ..... 4-22 to 25
Interface program 5-9

L

Line settings ..... 6-1,3; P6-1 to 7
check P6-1
create P6-4
List groups P2-9
Local area network 7-2
Lock files, UUCP 7-28
Lock unused logins 1-6

Log files, UUCP P7-17
.login file
..... 2-8
Login account 2-2
Login name 2-2
Logins
administrative 1-7
disable 1-6
system 1-8
UUCP P7-12
lost+found directory 4-25
lp command 5-42
lp login 1-8
lpadmin command ..... 5-7 to 27
lpforms command ..... 5-45 to 47
lpstat command 5-30,48
lpuser command 5-41

M

Mail programs 2-12
Maxuuscheds file 7-27; P7-13
Maxuuxqts file 7-27; P7-13
Message of the day 2-10
Messages, broadcast P2-11
Mount
filesystems 4-1; P4-2
forms 5-27,49
print wheel or cartridge font 5-27
Mountpoint 4-1
Move print requests 5-38
Move print requests in queue 5-42
Multiuser mode 3-6,7,8,10; P3-4

N

news command 2-11
News facility 2-10
Node name P1-3
nuucp login 1-8; P7-12

O

Operating system
- firmware mode 3-7,14
- multiuser mode 3-6,7,8; P3-4
- policies for changing run levels 3-1
- power up 3-10
- reboot 3-7,14
- remove power 3-14
- run levels ..... 3-6 to 14
- shut down P3-1
- single-user mode 3-6,7,13; P3-3

P

Partitions 4-2
passmgmt command 2-4
passwd command 2-4
Password aging ..... 1-2 to 6
- */etc/passwd* file 1-3
- */etc/shadow* file 1-3

Passwords 2-2
- administrative logins 1-7; P1-5
- aging ..... 1-2 to 6
- change 2-4
- system logins P1-6

Permissions file 7-6
- Add entries P7-11
- LOGNAME entry 7-18
- MACHINE entry 7-18
- setup 7-18

Permissions, default modes 2-8
Poll file 7-6
- Add entries P7-9
- setup 7-25

Poll remote systems 7-25
Ports, printer 5-13
Power down operating system P3-1
Power up operating system 3-10
Preen filesystems 4-20
Print queue 5-41,42

Print requests
accept 5-28,39; P5-5
banner page 5-23
change priority 5-43
move in queue 5-42
move to another printer 5-38
put on hold 5-43
reject 5-38
Print service 5-1
classes 5-25
commands 5-2
Default destination 5-26; P5-5
filters ..... P5-18 to 22
forms ..... 5-45 to 50; P5-11 to 17
interface program 5-9
print requests 5-42
printers ..... 5-6 to 28; P5-3
setup P5-3
start 5-5; P5-2
status information 5-30
stop 5-4; P5-1
troubleshooting ..... 5-32 to 37
Print specifications 5-24
Print wheels 5-14
Printers ..... 5-6 to 28
accept print requests 5-28,39; P5-5
add forms ..... 5-45 to 47
add to class 5-26
add to print service 5-7; P5-3
alerts 5-16,20,49
change configuration P5-6
character sets 5-14
classes 5-25
configuration 5-6
connection method 5-7
content type 5-11
default destination 5-26
delete form definition 5-48
delete from print service 5-27; P5-10
description 5-23
disable 5-29
enable 5-29; P5-5

faults ..... 5-20 to 22
font cartridges 5-14
forms 5-19
interface program 5-9
mount form or print wheel 5-27
name 5-7
port characteristics 5-13
print specifications 5-24
print wheels 5-14
priority level 5-41
priority limit 5-41
queue 5-41,42
reject print requests 5-38
status information 5-30
troubleshooting ..... 5-32 to 37
type 5-10
use as login terminal 5-8
use over a modem or network 5-8
users allowed access 5-22
Priority level 5-41,43
Priority limit 5-41
Profiles
user 2-8
Protocol, devices 7-11

Q-R

Queue, printer 5-41
Reboot 3-7,14
reject command 5-38
Remote systems
add to basic networking P7-7
call 7-25
remote.unknown file 7-27; P7-13
Restore from backup 4-18; P4-12
Restricted shell 2-9
root filesystem 3-3; 4-1; A-1
root login 1-8
Run levels ..... 3-1,6 to 14
change 3-12
firmware mode 3-7,14
multiuser mode 3-6,7,8

reboot 3-7,14
single-user mode 3-6,7,13

S

Schedule reminders, backup P4-14
Security 1-1
Set time and date P1-1
Shell program 2-3
restricted shell 2-9
shutdown command 3-14; P3-1,3
Single-user mode 3-6,7,13; P3-3
Start print service 5-5; P5-2
Start-up files 2-7
system 2-7
user 2-8
Stop print service 5-4; P5-1
stty command 6-6
su command 1-2
Superblock 4-7
check 4-22
sync command 4-19
sys login 1-8
Sysfiles file 7-6,26; P7-12
System log 3-2
System logins 1-8; P1-6
System name P1-3
Systems file 7-5
add remote systems P7-7
escape characters 7-17
setup 7-14

T

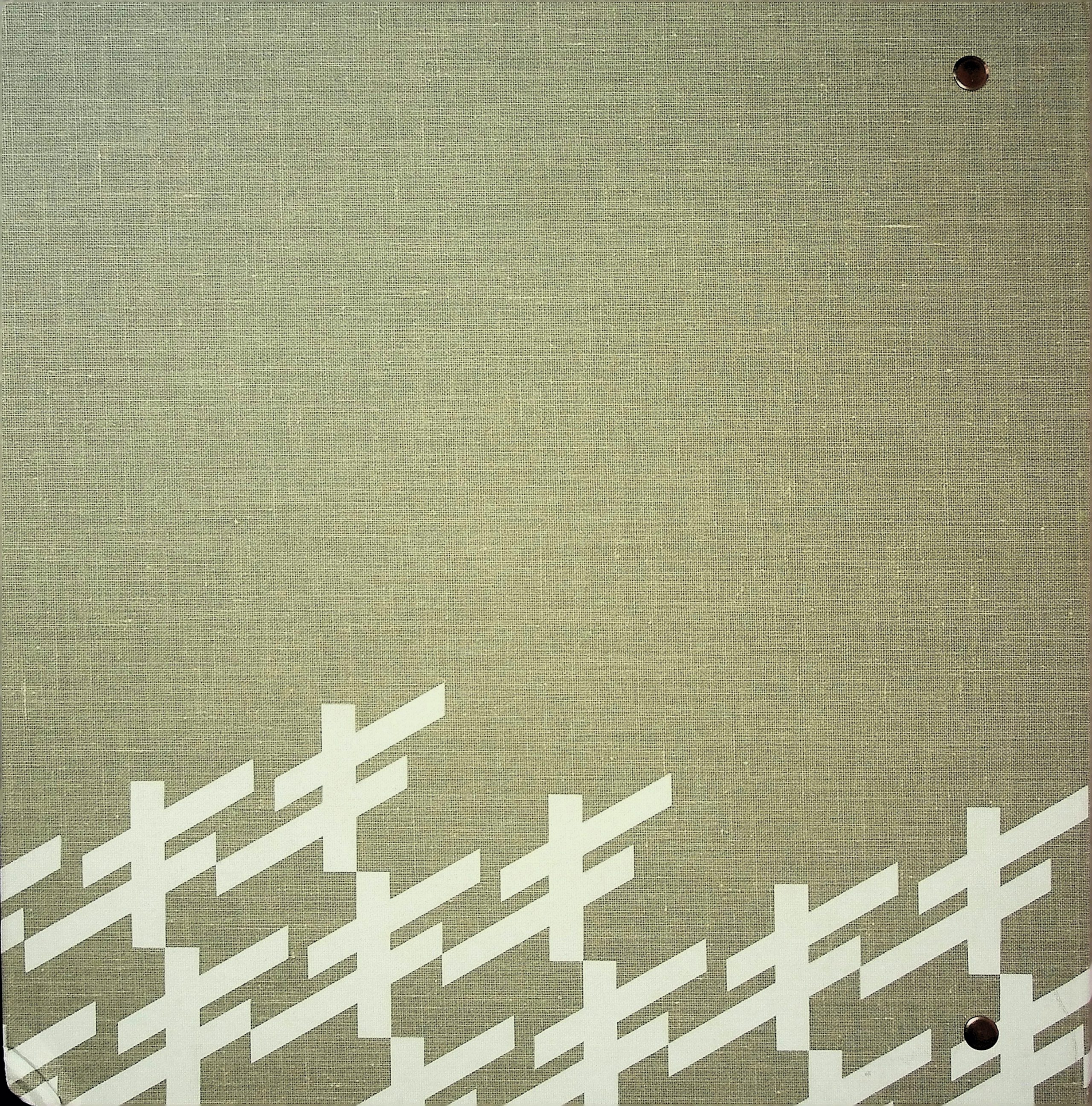
Temporary data files, UUCP 7-28
Terminals
alter configuration 6-4
options 6-6
specify type 6-5
Time, set P1-1
Tracks 4-4
Trouble reporting 2-12

TTY system 6-2
hunt sequence P6-4
line 6-1
line settings ..... P6-1 to 7
modify line characteristics 6-4
port 6-1

U

umask command 2-8
uname command P1-4
Unmount filesystems P4-2
User account 2-2
add P2-1
change entry P2-6
delete 2-4; P2-8
list P2-10
UID 2-2
Users
disk usage 4-14
setup P2-1
/usr/adm/sulog file A-8
/usr/lib/cron/log file A-9
/usr/lib/help/HELPLOG file A-9
/usr/lib/spell/spellhist file A-9
/usr/news directory A-9
/usr/spool/cron/crontabs directory A-10
uucheck program 7-4
uucico daemon 7-4
uucleanup program 7-3
UUCP
administrative programs 7-3
daemons 7-4
data files 7-29
debugging P7-18
Devconfig file P7-11
Devices file P7-3
direct links 7-30
execute files 7-29
file setup ..... P7-1 to 12
lock files 7-28
log files P7-17

logins P7-12
maintenance P7-14
Maxuuscheds file P7-13
Maxuuxqts file P7-13
Permissions file P7-11
Poll file P7-9
remote.unknown file P7-13
Sysfiles file P7-12
Systems file P7-7
temporary data files 7-28
user commands 7-2
work files 7-28
uucp command 7-2
uucp login 1-8
uudemon.admin shell script P7-15
uudemon.cleanup shell script P7-16
uudemon.hour shell script P7-15
uudemon.poll shell script P7-14
uugetty program 7-5
uulog program 7-3
uupick command 7-3
uusched daemon 7-5
uustat command 7-3; P7-15
uuto command 7-2
Uutry program 7-4
uuxqt daemon 7-4

V-W-X-Y-Z

Variable, environment 2-6
VTOC (volume table of contents) 4-2
wall command 2-11; P2-11
Work files, UUCP 7-28